HUMAN-LEVEL NATURAL LANGUAGE UNDERSTANDING: FALSE PROGRESS AND REAL CHALLENGES

By

Perrin G. Bignoli

A Dissertation Submitted to the
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements of the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Cognitive Science

Approved by the
Examining Committee:

_____
Professor Nicholas L. Cassimatis
Thesis Advisor

_____
Professor Selmer Bringsjord, Member

_____
Professor Ron Sun, Member

_____
Professor Michael Lynch, Member

Rensselaer Polytechnic Institute
Troy, NY

November 2013
(For Graduation December 2013)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

There are a number of people who have been instrumental in helping me to complete the work that is described in this doctorate dissertation. Of course, a huge host of individuals have influenced me throughout my life that must remain anonymous here for the sake of brevity. It is my sincere hope that they can draw some satisfaction from the fact that their friendship, time, effort, encouragement (and even antagonism) have all led, even in an indirect way, to what I would like to think is a contribution to science and the foundations of what may be a successful research career. I am so privileged and humbled to have known you all.

My odyssey to Troy and RPI began with the dedicated support and attention paid to me from my childhood until the present by my parents, Mr. Jerome H. Bignoli and Mrs. Kathleen S. Hill, my sister, Mrs. Callan M.R. Bittrich, my stepfather, Mr. Steven R. Hill, Sr., and my maternal grandparents, Mr. Gregory M. Leska, Sr. and Mrs. Ruth Leska. I am also indebted to the seemingly unbounded sequence of incredible teachers in the Johnson City Central School District and professors at Cornell University and RPI who have provided essential guidance in my education and personal intellectual development. Likewise, I owe much to the people who I came in contact with during my employment at the Air Force Research Laboratory in Rome, NY. Without them, I most likely would never have made the decision to enroll as a full-time PhD student at RPI. Dr. Paul Bello, who has since become a major figure at the Office of Naval Research, recognized my interest in Cognitive Science and Artificial Intelligence and introduced me to my advisor and his research group in RPI's Department of Cognitive Science.

The past and present members of that research group, the Human-Level Intelligence Laboratory (HLIL), have all been invaluable sources of inspiration and ideas. One could not hope for more dedicated colleges and collaborators. I am also deeply grateful for the help I received from the members of my doctoral committee, Professor Selmer Bringsjord, Professor Michael Lynch, and Professor Ron Sun, as well as the many administrative and support staff members at RPI that helped me throughout my time as a student there. Of course, none of this research would have been possible without the

enormous guidance and support of my advisor, Professor Nicholas Cassimatis, who founded the HLIL Research Group and the SkyPhrase Corporation.

ABSTRACT

The field of Natural Language Processing (NLP) focuses on the study of how utterances composed of human-level languages can be understood and generated. Typically, there are considered to be three intertwined levels of structure that interact to create meaning in language: syntax, semantics, and pragmatics. Not only is a large amount of language-specific information required to learn syntactic patterns and semantic constructions, but also an even vaster amount of commonsense and world knowledge is required to interpret language at the pragmatic level. Because the task of creating a system that is fully capable of processing language at the human level of proficiency is extremely daunting, researchers have mostly focused on developing constrained models of specific aspects of NLP, such as Part-of-Speech tagging or named entity resolution. However, a number of more comprehensive NLP platforms, produced by companies such as IBM, Google, Microsoft, and Apple have taken on the task of providing fully functional, natural-language-driven services.

Based on the level of success reported by researchers, as well as from the advertisement campaigns of companies that deal with NLP technology, it would seem that the NLP problem is well on its way to being solved by the techniques currently employed by the field. Currently, knowledge-lean, statistical, machine-learning like approaches dominate the NLP research arena. These approaches have largely displaced earlier knowledge-rich, expert-system like approaches, because of their increased flexibility and robustness. Because NLP is widely considered to be an "AI-hard" problem, the resolution of the challenges of making a human-level NLP system would have huge implications for both Cognitive Science and AI in general.

Unfortunately, it turns out that much of the present optimism in the NLP field is essentially hyperbole. An in depth analysis of knowledge-lean NLP techniques will demonstrate that they are significantly underpowered for handling tasks such as basic syntactic parsing, let alone full natural language understanding. To make matters worse,

the NLP field has been led down this path in large part due to the kinds of evaluation methods used to gauge success on individual NLP tasks. Once these metrics are analyzed, it is straightforward to see how they can create confusion and misleading results. In light of these findings, it is the intent of this document to demonstrate the case for modifying the current knowledge-lean NLP culture with techniques that are derived from Cognitive Science.

1. Literature Review

1.1 Introduction

This literature review aims to provide an in-depth assessment of Natural Language Processing (NLP) techniques in an effort to critically examine the direction of progress in that field and to suggest new directions for the work of elevating the linguistic capabilities of artificial systems to the level exhibited by humans. Additionally, this review makes the case that the study of the current state-of-the-art of NLP techniques is extremely relevant to the field of Cognitive Science. A brief introduction is provided to frame the development and analysis of NLP techniques, which is a pursuit which often framed as an activity in the realm of Artificial Intelligence (AI) research, in a way that makes clear why Cognitive Science stands to benefit from such an endeavor. Furthermore, a novel assessment of the common successes and failures of these techniques is performed in order to obtain a clearer view of the prevailing trends in the NLP domain.

As with all branches of AI, there is an internal schism between the general types of approaches used to solve NLP problems. For the purposes of this document, all NLP work is categorized as belonging to one of two families of approaches: 1.) the knowledge-rich approach and 2.) the knowledge-lean approach[1]. At the present time, a majority of NLP researchers is convinced that the knowledge-lean approach is superior to the knowledge-rich approach in terms of power, feasibility and scalability (Brill & Mooney, 1997) (Manning & Schütze, 1999) (Jurafsky & Martin, 2000). Prominent exponents of this opinion have claimed that there is a clear advantage of statistical approaches over knowledge-rich ones, and moreover that the statistical approach is the only one that could possibly be successful (Norvig P. , 2011). Some of these scientists have taken the unprecedented additional step of declaring that the era of rationalist science has concluded and that it is now more productive to obtain theoretical frameworks directly from the empirical analysis of very large data sets (Scheutz, Cantrell, & Schermerhorn, 2011). Because there is concrete evidence to support this line of

[1] Note that there are other common names for methodologies that fall in the "knowledge-lean" paradigm. Some of the more commonly encountered names are: machine-learning, data-driven, and statistical.

reasoning in the form of reported successes in developing practical, large-scale, high-complexity, real-time NLP platforms, a great amount of credence is warranted to considering this review for dictating the future of the field. Indeed, the evolving linguistic capabilities of computers backed by statistical AI techniques seem to speak for themselves.

Even though there is an increasing tendency to discount other avenues of approach towards solving NLP problems and thus placing ever more resources and attention in the development of statistical NLP techniques, a more pragmatic survey of the field reveals that the purported successes of this program have been overstated. It is notoriously difficult to predict the best future direction of research for any given scientific discipline from the prevailing trends observed in that discipline (Board on Behavioral, Cognitive, and Sensory Sciences (BBCSS), 2006) (Kuhn, 2012) (Bignoli, 2012). Researchers often espouse opinions that are conditioned on their background, training, and past work. This pattern of behavior is expected when any kind of expertise is involved in working towards a complex goal. However, when such inertia begins to detract from achieving these goals, a fresh path is required to move forward.

A deep probe into the end results of statistical NLP systems reveals that their performance is extremely limited, that they are difficult to extend or generalize to novel domains, and that they are not likely to be able to overcome their shortcomings simply through the availability of larger data sources and more computational resources (c.f. Moore's Law (Moore, 1975)). Examining the metrics employed to gauge the effectiveness of individual NLP technologies on narrow tasks illustrates that, despite many claims to the contrary, modern NLP is still primitive. Essentially all of the performance assessments commonly used by NLP researchers give the appearance of far greater ability than effectively is the case. One primary cause of this situation is that because the task of creating a single system of sufficient complexity to handle the processing of large segments of real-world utterances is still beyond the limits of current technology, it has been necessary for most researchers to focus on small amounts of isolated text or a particular sub-problem that can be teased apart from the challenge of total comprehension. Thus, it has become standard practice to report results that are not only misinforming, but also that bear little consequence for those interested in developing

true human-level NLP applications. Although some work has been done to model human-like language use at the full human level of complexity (Schuler, Wu, & Schwartz, 2009) (Turing, 1950), these efforts are still in their nascent stages and are far outnumbered by the much more restrictive projects that dominate the attention and resources of the NLP field.

An additional factor that confounds attempts to formulate a realistic assessment of state-of-the-art NLP technologies is the inflated level of over-confidence that companies produce surrounding their NLP-capable product lines and research initiatives. Alan Turing, one of the founders of computer science and an early speculator on the development of intelligent machines, opined that it would be possible to determine if a device was capable of sophisticated cognition by proxy via observing its ability to interact with a human using natural language. This procedure became known as the Turing Test (Hutchens, 1996) and has remained at the forefront of how popular science defines the intelligence-levels of AI applications.

Since natural language is so tightly tied into the popular view of what characterizes human intelligence, it is a popular pursuit of technology companies to develop products that at least give the appearance that they can understand language. Paradoxically, faking competency in language use without doing much work to advance the state of the field is one of the easier tasks for an AI researcher (Weizenbaum, 1966). Even the simplest program, when placed in the proper contextual environment and subjected to appropriate restrictions, can dupe humans into believing that they are interacting with an intelligent agent. Some of the most famous examples of "zero-knowledge" systems (e.g. ELIZA, a type of program called a chatbot, which operates largely by responding to conversation by permutation of the input text (Mauldin, 1994)) were first produced using extremely little linguistic sophistication and could also run on the limited computing machines of almost half a century ago. However, as recently as the late 1990's, such zero-knowledge approaches (e.g. PC Therapist (Halevy, 2009)) were serious contestants in the Loebner Prize contest (Loebner, 1993), which is the premier incarnation of the Turing Test. Much of this apparent success with knowledge-lean techniques can be attributed to the human inclination to anthropomorphize the behaviors of agents (Hofstadter, 1996).

Although it is possible to write programs that use tricks and boilerplate responses to trick humans in some situations, more in-depth interaction, such as the understanding of domain- general commands, is beyond the capacity of any existing system. Most applications that attempt human-level NLP have performed so underwhelmingly that the bar for comparative success is quite low. As with the field in general, comparatively shallow statistical methods dominate the commercial and corporate science NLP projects, which are nevertheless promoted as significant milestones towards the achievement of human-level intelligence. An examination of some of the more the prominent projects of this type is presented in Chapter 7 to demonstrate how limited these systems truly are.

Once a more accurate perspective is taken on the current trends in NLP methodology, it is possible to begin to investigate the true topology of the field and to reveal the strengths and weakness of the most popular approaches. As a result of this type of analysis, new recommendations for more promising future directions for NLP research can be formulated. However, this movement towards reformation should not be taken to imply that all efforts in this area have been misguided or that absolutely no contributions have been made towards a solution to human-level NLP. There are many aspects of on-going NLP research that are extremely valuable and informative and will surely play a prominent role in whatever systems ultimately achieve human-level performance. To understand this point, consider the following metaphor: scientific progress is often conceptualized as a pendulum that swings back and forth between different extremes; NLP was once dominated by knowledge-rich and deductive-logic-based approaches, but a number of factors led to the supplanting of such research by knowledge-lean frameworks. The important lesson to learn is not that knowledge-lean approaches are useless or wrong, but that they are, in fact, are just subject to their own variety of limitations. Moreover, knowledge-lean methodologies are actually very closely related to the knowledge-rich methods that they purportedly would replace. It would be expected that, eventually, the pendulum will be swinging back towards the knowledge-lean approach. The main intent of this dissertation is to argue for this change of perspective, but to also remind its readers that it is critical to synthesize the best aspects of the knowledge-lean and knowledge-rich paradigms without selecting one type of methodology exclusively over the other.

While good reasons exist for researchers to devote effort and resources to the continued development of knowledge-lean, machine learning techniques—increased flexibility and robustness (Hernandez-Lobato, 2010), reduction of hand-coding solutions to narrow problems (Cardie, 1996), and creation of efficient learning algorithms for complex patterns (Salakhutdinov & Larochelle, 2010)—there are numerous reasons not to entirely abandon knowledge-rich approaches to NLP. Many problems that commonly arise during language understanding are inherently knowledge-rich processes. For instance, reference resolution and anaphora are known to require large amounts of structured reasoning to solve (McShane, 2009), which is why the standard tasks for assessing knowledge-lean approaches are often tightly constrained (e.g. as an illustration consider the guidelines put forth for participants in the Message Understanding Conference (Chinchor, 1998)). This type of constraint results in the evaluation of NLP research on what essentially amounts to toy problems that do not realistically represent the demands of real-world, human-level language processing. Once this fact is understood, the initially impressive performances of many knowledge-lean NLP methods are revealed to be quite distant from the final goals of the field.

In summary, this literature review is designed to focus on the unique role that the drive towards developing human-level NLP plays in the intersection of AI and Cognitive Science, which will help to frame the remainder of the dissertation. A brief overview of the contents of these other sections follows. By critically analyzing both the leading academic approaches in the NLP field and the most mature commercially-oriented technological offerings that the two families of NLP approaches have produced, a comprehensive picture of the field is established. The metrics used by NLP researchers for a variety of tasks will be examined and found to be poor assessments for obtaining the ultimate goal of human-level AI. A detailed look at a select number of NLP tasks and methodologies, namely those associated with Part of Speech Tagging and Syntactic Parsing, serves as a window into the general shortcomings of the current era of knowledge-lean NLP. Finally, an outline for future research that is more in-line with the realization of human-level NLP is presented, with a particular focus on how to exploit the deep underlying relationship between the apparently disparate methods used by knowledge-rich and knowledge-lean approaches to AI.

1.2 Applying Artificial Intelligence Methods to Cognitive Science

Cognitive Science and Artificial Intelligence (AI) are deeply related disciplines. On one hand, Cognitive Science aims at uncovering the mechanisms and properties of existing systems that are capable of intelligent behavior. On the other hand, AI investigates how to implement new systems that are capable of intelligent behavior. Since there are many possible definitions for what constitutes intelligent behavior (Legg & Hutter, 2007), these definitions are necessarily vague. At present, there are three types of systems that are generally accepted to exhibit some kind of intelligence: 1.) humans, 2.) non-human animals, and 3.) computing machines. Of these, the first two are naturally occurring, whereas the third must be produced by artificial means. Traditionally, Cognitive Science has restricted itself mostly to the realm of natural intelligent systems--with a dominant focus on humans—although it is conceivable that, in the future, artificial systems will be sufficiently complex to warrant direct study by cognitive scientists. For now, the task of evaluating the mechanics of artificially intelligent agents falls in the domain of AI research.

Despite this traditional division between natural and artificial systems, it is imperative that Cognitive Science and AI endeavor to support each other. AI stands to benefit from results in Cognitive Science, because an understanding of the operations of existing intelligent systems would almost certainly be critically important for the development of novel ones. Considering the logistic difficulty of implementing an information-processing device with the power of the human brain with the limited physical resources that nature devotes to it, there is reason to believe that, even if there are multiple engineering solutions to this problem, the difficulties involved would be cause to utilize as much of this available blueprint as possible. It has been emphasized that basing AI technologies on a human-inspired cognitive substrate is a sound practice for researchers (Cassimatis N. L., 2006). Nevertheless, there are a number of caveats in attempting to make this transfer of knowledge from results in empirical psychology and neuroscience: most notably that individual differences between experiment participants, as well as noise from uncontrolled irrelevant factors, limits the specificity of such insights for the present time (Cassimatis N. L., 2012). Rather than attempt to copy the singular

idiosyncrasies of the human cognitive system, such efforts would be better applied to high-level, abstracted processing mechanisms and representation schemes.

This potential for fruitful crosspollination has not been lost on many of the leading figures in both Cognitive Science and AI—many projects have been launched that leverage results from both fields (Luber, 2011). Moreover, AI is often considered to be one of the methods available for Cognitive Scientists to pursue their goal of understanding cognition (Thagard, 2012). Cognitive Science is driven largely by experimentation, which, on the surface, distinguishes it from AI and Computer Science. Nevertheless, a compelling case can be made that individual programs and projects serve as the analog for empirical methods in AI research (Simon, 1995). Despite these important connections, a barrier often is erected that cordons off the technical, computational, and quantitative methods in AI from direct consideration by Cognitive Science. The idea that the abstract patterns and algorithms of AI researchers could be considered as tools for standard Cognitive Science is at best, kept at the fringe of the field.

There are, of course, numerous other types of manifestations of Cognitive Science research that overlap significantly with other fields, such as better design of artifacts (Norman, 2002), enhancements in the effectiveness of communication (Smith & Swinyard, 1988), and insights into new therapies for psychopathology (Strickland, 1997). Each of these areas, as well as many others that also seem to be acceptably classified as being in the mainstream of Cognitive Science work, requires a high degree of technical proficiency in disciplines as wide-ranging as: ergonomics, systems engineering, communication, media and public relations, medicine, economics, and clinical psychology. These diverse technological advances serve as direct justifications of why corporations, universities, and government agencies should continue to fund additional research in Cognitive Science, as there are concrete rewards to be had from such study. Cognitive Science is not just involved with abstract wonderings about the nature of humanity, but it also is a vehicle for bringing about human-centric refinements to technologies.

While the fruits of AI are also highly visible and many of its practical "end products" can be shown, in part, to be derived from work in Cognitive Science, it is often

easy to overlook these contributions, as they tend to be seamlessly folded into the technologies they improve (c.f. the disconnect between public opinion about the progress of AI and the work actually performed by experts (Dodd, Grant, & Seruwagi, 2011)). Few lay people realize that something as apparently rudimentary as an Internet search service is, when evaluated fairly, a great achievement that is the culmination of decades of AI research. It is difficult to identify areas of inquiry and modes of human endeavors that progress in AI is unable to augment in some way—just as it is difficult to imagine a human institution or behavior that is opaque to the investigations of cognitive scientists. A final component of the argument for couching the mathematics and engineering aspects of AI within the scope of Cognitive Science is that there is evidence that the human cognitive system truly does utilizes processes that are highly similar to automated reasoning mechanisms (Cassimatis et al., 2010). Thus, there should be due cause for the submission of a technical AI-oriented dissertation to a Cognitive Science department.

Additional perspective can be obtained by considering more precisely how methods in AI are able to directly impact progress in Cognitive Science and how findings in Cognitive Science could serve to guide the direction of AI. The practice of developing cognitive models in order to propose precise mechanisms to explain empirically-studied individual human behaviors and to test higher-level theories about the functions required to support cognition is perhaps the most prominent example of this exchange. A cognitive model is traditionally created by formalizing a particular theory of human cognition, which may have essentially any scope in any domain of human behavior. For instance, computational models of human memory, perception, motor action, speech, and reasoning have been produced at many levels. By comparing the properties of these models with data from human trials, it is possible to provide a degree of insight into whether a particular review for the mechanism behind such behavior is likely to be accurate. However, work at this fine-grained level of detail is tantamount to "playing 20 Questions with nature," which is a decidedly unwinnable game (Newell, 1973).

It is essential to abstract beyond isolated individual phenomena and produce frameworks for capturing the general trends of the human cognitive system that can be utilized to create families of models with common theoretical structure. Such frameworks are known as cognitive architectures. Some of the most developed cognitive

architectures are ACT-R (Anderson J. R., 1996), Soar (Milnes et al., 1992), EPIC (Meyer & Kieras, 1997), CLARION (Sun R. , 2003), ICARUS (Langley & Choi, 1999), and Polyscheme (Cassimatis & L., 2001). However, it is imperative to use these frameworks to explore beyond the fitting of model parameters to laboratory data. By exploring various algorithms and representations for reasoning within a cognitive architecture, it is not only possible to refine theories about human cognition—for instance, a particular hypothesis might be revealed to be computationally implausible—or to produce more efficient automated reasoning procedures—for instance, a model of human memory might be used to implement a faster search program. From a pragmatic standpoint, a compelling case can be made for defining the desired outcome of AI research to be computer programs that "exhibit behaviors that would be regarded intelligent if they were exhibited by human beings" (Simon & Kaplan, 1989).

There has also been a body of work devoted to uncovering the formal rules of thought that humans obey when solving real-world problems. Due to the vast amount of information, the potential for high cost errors, and demanding time constraints of real-world reasoning problems, the human cognitive system, and its entire hereditary lineage, evolved to enable it to solve the problems most integral to increasing individual fitness in the most efficient ways possible. Work has been done to characterize the information processing heuristics that the human cognitive system brings to bear on the difficult problems that it encounters (Gigerenzer, Todd, & Group, 1999). These strategies are often at odds with some of the traditional wisdom of the machine learning field. For instance, an underlying assumption that machine learning practitioners seem almost universally to make is that more the information about a problem that can be fed into an algorithm, the more accurate the performance of the algorithm will be. Doing so blindly often lead to overfitting; it is an open problem to determine the set of all problems for which additional information improves the eventual outcome. Experiments have demonstrated that humans do better with limited amounts of information on some types of tasks (Goldstein & Gigerenzer, 2002). Further studies have demonstrated that one of the more plausible architectures for aspects of human cognition is that some component of it is strongly reliant on a relatively small number of systematic biases and bets about problem structures commonly encountered in the physical and social environments. Of

course, human cognition extends beyond the scope of such heuristics, but its capacity to do so appears to be constrained by the availability of sufficient cognitive resources and appropriate circumstances for its deployment (Kahneman, 2011).

Additionally, the operation of the human mind has been analyzed from a physiological perspective, according to the general computational properties of its components exhibit. At the broadest level, the similarities and differences in the functional properties of the brain's hemispheres have been identified with distinct aspects of cognition (Ornstein, 1997). The left hemisphere tends to examine details in isolation from other concepts, is capable of suppressing alternative interpretations, and is configured for sequential processing. On the other hand, the right hemisphere is more involved with integrating various facets of a concept together, considering multiple alternatives, and performing more global computational processes. These divergent properties, however, do not lead to a strict division of labor in the brain, in which one hemisphere is wholly or nearly wholly responsible for a given cognitive task. Rather, each hemisphere is now thought to contribute in its particular way to essentially all cognitive activities, including language, which long was held to be the demesne of the left hemisphere. Indeed, the right hemisphere has been shown to activate alternative interpretations of linguistic elements—a capacity that would be extremely useful for handling the high level of ambiguity in natural languages. Furthermore, this discovery may lead to subsequent research that would illuminate more of the specific algorithms that the brain uses to parse natural languages. At a finer level of structural detail, a number of projects have been proposed to develop neural-network-like architectures that are based on the actual physical organization of the human brain. Most of these approaches have focused on a number of regions of the neocortex, which all seem to exhibit the same type of underlying structure. There are several different interpretations of how this structure might be coopted into an information processing algorithm. Some notable examples include Hawkins' Hierarchical Temporal Memory system (Hawkins & Blakesless, 2005) and bijama (Menard & Frezza-Buet, 2005).

The above examples demonstrate that the study of Cognitive Science benefits tremendously from constructing metaphors from AI research that help examine the human mind in its role as an information processing system. But cognitive science can

also directly lead to advancements in AI (Potapov, Rodionov, Myasnikov, & Begimov, 2005) (Bringsjord & Clark, 2006). For certain types of problems, the heuristic processing biases of the human cognitive system can be successfully adapted to machine reasoning problems. For instance, one of the main problems with learning by induction (in the machine learning sense) is that there are an infinite number of solutions for fitting data with some explanation. In domains in which humans are adept reasoners, it is possible to examine the constraints that the human cognitive system inherently imposes on problem instances of that domain. Although humans may not achieve perfect accuracy, many of the domains in which humans function do not require perfection. The fallacy that advanced reasoning must be optimal has lead AI down a "garden path" of intractability. Satisficing, which entails finding a solution that is sufficiently acceptable according to some predefined measure, is a more computationally feasible function than is optimization (Simon, 1956).

Alternatively, the study of the goals of humans and the salient components of the environments in which humans interact can be leveraged to inform the design of artificial systems that are useful tools for humans (Simon, 1996). Representations that have been developed by humans to solve difficult problems can be employed to produce efficient AI algorithms. It has been famously stated that finding the right representation is the most difficult aspect of many problems; once a good representation has been established, the procedure for obtaining solutions in that form is often straightforward (Simon, 1996). However, it is vital to provide artificial reasoners with this heretofore human capability for creativity and adaptability in problem solving (McCarthy, 2007), thus enabling AI systems to generalize beyond a human-defined and managed "bounded informatic situation." Furthermore, the same representations that aid humans in solving problems may be useful for the design of artificial systems (Norman & Dunaeff, 1994). Yet another parallel between AI and Cognitive Science will probably become more vital to explore as artificial systems increase in complexity: just as Cognitive Science conducts experiments to uncover the nature of the human cognitive system, it is possible to cast Computer Science as a discipline that empirically uncovers the nature of computation for all symbolic processing systems (Newell & Simon, 1976). Presumably, at least some of

the techniques that have been developed for empirical Cognitive Science research could be applied to empirical Computer Science studies.

1.3 The Relevance of NLP to Cognitive Science

Even though there is ample reason to study the all of the insights that AI algorithms could shed on human cognition, it is beneficial to instead devote a deep examination of one particular pursuit of AI research. Perhaps the best candidate for this is NLP research. Two principal reasons stand out as motivators for selecting NLP as a center of focus. First is the clear fundamental importance that language plays in human activity; high linguistic capacity is one of the more strongly characterizing cognitive components of the human species. Thus, language processing should be an area that Cognitive Science is deeply concerned with exploring. Second is the overwhelming evidence that language use is intertwined with the entire gamut of cognitive processes. Although it is difficult to find any cognitive domain that cannot be somehow related to many others, language overlaps with all of them to an incredibly large degree. There is no doubt that any increase in the understanding of how humans process language would lead to an increase in understanding how human cognition operates as a whole.

Humans have always had a special fascination with the faculty of language. Out of all the characteristics that seem to separate humanity from the rest of the animal kingdom, the use of language is among the most prominent. Even though, at least through an anthropomorphizing lens, many species of animals engage in intelligent or what is deemed to be "human-like" behavior, no other animal species approaches the communicative power of humans. A large number of animals can be conditioned to respond to language, delivered both in spoken (Jensen, 2007) and written (Yellen, 2009) forms, but it is difficult to show that this capacity extends beyond sensory discrimination. Highly-trained individuals of some types of animals, such as apes (Thompson, 2010), dolphins (Herman & Uyeyama, 1999), and some species of birds (Abe & Watanabe, 2011) can apparently transcend the boundaries of mere conditioning and seem to be capable of understanding, and in some cases, generating language that obeys some set of grammar rules. Yet, despite these superficially astounding feats of language, the verbal behavior of humans truly makes the species unique on Earth. The size of the average

human vocabulary dwarfs that of even the most linguistically fluent animals; humans can communicate about novel topics in nearly any kind of situation or environment[2].

Moreover, cutting-edge neuroscience has revealed that there are major differences between the microscopic structures of human brains and the brains of other animals, even those most closely related to humans (Human, 2007). This finding may help overturn the presupposition in place at least since the time of Darwin that humans are merely simple extensions of animals. At least in terms of the fine structure of the CNS, the extension hypothesis would appear to be false if the aforementioned research is valid. Other research along the lines of distinguishing human neurophysiology from that of other animals has indicated that while both humans and other primates have groups of cortical cells, commonly called "mirror neurons," they are applied to more complex tasks in human cognition. These mirror neurons are activated when an individual performs the same action that it is observing another individual complete. However, in humans, unlike in monkeys, there are mirror neurons that decode how an action is performed rather than neurons that simply recognize similar actions (Rizzolatti, 2005). It has been hypothesized that human language may have developed as an exaptation of the mirror neuron system, which may have originally been used to decode gestures and other complex actions (Skoyles, 2000). Furthermore, although many animals have cortical regions that are analogous to the speech processing areas of the human brain, for instance, Broca's area, in humans these regions are comparatively far more developed and integrated into the rest of the cerebral cortex.

Although there is no doubt that even the most linguistically able animal commands a minimum of proficiency compared with the repertoire of the average human child, it should be noted that linguistic capacity in species seems to correlate with development in complex social and physical environments (Slobodchikoff, Perla, & Verdolin, 2009). The fact that all forms of natural language play out in such a knowledge-rich tableau, with numerous complex relationships, emphasizes the role that cognition plays in the workings of natural languages. Thus, by studying how humans understand and generate language, much insight can be obtained not only into many other cognitive mechanisms, but also how those mechanisms are integrated to produce

[2] Clearly, this is restricted to the set of environments that permit communication.

phenomena as complex as natural language (Meteyard & Vigliocco, 2008). Sensory perception (McClelland & Elman, 1986), motor control (Ackermann & and Riecker, 2004), emotions (Gendron, Linquist, Barsalou, & Barrett, 2012), mental imagery (Barsalou, 1993), pattern matching (Jackendoff, 1993), and executive control (Gruber & Goschke, 2004) all play a role in the cognitive substrate for language.

Yet, language is more than just a composite of cognitive mechanisms, but rather a holographic system that emerges from those components (Pinker & Jackendoff, 2005). Therefore, there is a vital link between general cognition and language, since language plays a central role in human social interaction and individual behavior (Bickerton, 2996). To accept this, one does not need to subscribe to the long-discredited "Sapir-Whorf Hypothesis" (Sapir, 1921) (Whorf, 1942) that an individual's language unconditionally shapes the concepts that they bring to bear when reasoning about the world. Rather, language is probably better thought of as a view into the mind that provides a limited-dimensional projection of the total conceptual space of a speaker. There are a large number of books that have contained comprehensive criticisms of the Sapir-Whorf Hypothesis; one of the most notable of these is "The Language Instinct" by Steven Pinker (Pinker, 1994). By establishing a proper perspective on the interaction between linguistic cognition and cognition as a whole, it is easier to plot out future research that aims to produce models for NLP. In particular, the place of language in the total cognitive framework should suggest that while there probably is some actual division between linguistic and non-linguistic knowledge, these two systems are much more intertwined than they are separate and that it would be fruitless to produce models for one without considering the other.

There are other misconceptions about the relationship between language and cognition that must also be addressed in a comprehensive review of the subject. For instance, it was once the case that some researchers strongly believed that language somehow existed in a dedicated, specialized, inherent, separate "module" from other cognitive processes. The Universal Grammar expounded by Chomsky (Chomsky, 1986) is one incarnation of this type of theory. Instead of being learned from a blank state

through some kind of conditioning process,[3] Universal Grammar casts language acquisition as the process of setting a few universal parameters that characterize specific languages. The argument from the poverty of the stimulus claims that there is not enough supervised feedback provided by in the linguistic learning environment to uniquely induce a grammar with the complexity of natural languages (Gold, 1967). Alternatively, a module-based view of cognitive mechanisms, including those that governed linguistic processing, was proposed by Fodor (Fodor, 1983) and thereafter became a popular conceit in Cognitive Science and AI research. This hypothesis differs from Chomsky's theory in that each Fodor-type module has access to a limited amount of domain-specific cognitive information rather than the entire store (Okasa, 2003).

In light of contrary evidence (Rumelhart & McClelland, 2986), the strict modular view of language and the mind has fallen out of favor for more integrative theories. However, there is still some dispute over how dissociated language is from the rest of cognition. Some researchers have thought that the neural mechanisms underlying the language faculty, despite being extremely complex and multimodal, could actual function more or less independently from the rest of the cognitive system. One of the major lines of evidence that language is an independent system has been the discovery of certain developmental pathologies that devastate general intelligence while leaving linguistic proficiency intact. Two of the most intensely-studied medical conditions that lead to the devastation of cognition with the exception of language use are William's Syndrome (Bellugi, Wang, & Jernigan, 1994) and Hydrocephalus (Cromer, 1991). However, more recent research has demonstrated that much of the evidence that was initially interpreted as supporting the separation of linguistic and non-linguistic cognition can be recast as showing that there is not a separation at all, but rather a tight integration (Sacks, 1995).

There are even broader arguments for the integrative view of language and general cognition from neurological studies. Patients that have lost their ability to perceive a particular aspect of a sense, such as color, appear to lose their ability to name colors after a long span of achromatopsia (Sacks, 1995). It is difficult to see why this would be the case if language was somehow innate or fully encapsulated from the

[3] N.B. Behaviorism (Skinner, 1938) was the prominent approach in Psychology at the time that Chomsky was writing

sensory processing systems. Furthermore, regions of both hemispheres of the brain are involved with the processing of language (Ornstein, 1997); these regions also serve non-linguistic purposes, which casts doubt on whether there are any mechanisms in the brain that are solely tasked with language processing. Broca's area, long since thought to be one such language processing region, actually is implicated in many tasks that require sequential processing of discrete events (Katz & Lin, 2002), which include understanding of (Fadiga et al., 2006) actions and interpretation of gestures.

In terms of non-linguistic aspects of cognition, it has long been observed by etiologists that while other species are clearly capable solving problems in many of the same domains as humans, there is, at the very least, a huge gap in the degree of proficiency between these two groups. Most relevant to this review, then, is what follows from these observations. If there are actual differences between humans and animals, which occurred over a relatively short span of evolutionary time, then it would presumably be possible to identify which aspects of human reasoning are distinct from non-human cognitive behavior and to then exploit those traits as computational mechanisms. Since human cognitive adaptations appear to be fairly specific and structured (Gigerenzer & Selten, 2001), it would stand to reason that a dedicated knowledge-rich system could potentially emulate this facet of human cognition. Such an approach can be made within the framework of developing cognitive architectures for specific domains, which can use statistical techniques to extrapolate to new areas and to cross-pollinate between existing knowledge bases. The systems that result from the implementation of such systems would undoubtedly be a crucial step towards the goal of true human-level NLP.

Holding the privileged place that it does in the realm of biology, language was once used as a benchmark metric for assessing the intelligence of an individual directly (Murphy S. , 2012). Although the early efforts to equate language skills with high general intelligence were misguided (Przybyla & Zhou, 2008), this practice reveals how essential language seems to human thought to lay people and experts alike. When scientists and engineers alike peer into the mechanics of human cognition, many of them imagine that there is a language of thought (Church & Rau, 1995) (albeit distinct from languages of discourse) and use this metaphor to guide their efforts. An examination of

the evolution of this trend in the fields of Cognitive Science and AI reveals that there is a more formal interrelationship between logic, language, and conceptual thought (Murphy S. , 2012). Philosophers have employed logic to formally capture the content of minds and to study how they perform acts of inference (Przybyla & Zhou, 2008). Indeed, logic has successfully been used to model conceptual knowledge, both in terms of producing the type of abstractions of the structure of reality that are the fundamental components of most computer programs and in terms of how concepts are implemented in the human mind. This should be taken as ample evidence to support the use of logic-based methods in Cognitive Science research. While non-linguistic representation and processing underlies a great deal of cognitive activity, for almost all humans, this knowledge is accessible and manipulable to some degree through natural language. Additionally, the study of logics and representations has led to notions of formal syntax and semantics, which have counterparts in the study of natural languages. It is therefore natural for the information expressed in the language of the mind or communicated through natural language to be cast in propositional form (Church & Rau, 1995). This parallel has in part led to the inception of knowledge-rich and symbolic AI. For any science that endeavors to study the overarching cognitive architecture of the human mind, the study of language must be a vital component.

Furthermore, the study of language has other ends than fostering a better understanding of the human condition. Since the inception of the fields of Psychology, Computer Science, Information Science, and Cognitive Science, there has been a long-standing ambition to employ technology to improve human performance and competency in essentially every field of endeavor. Moreover, the desire to produce more intelligent machines and thereby a society with more intelligent institutions, has led to an explosion in interest in developing systems for automating human-level natural language processing. Semantic Web technology, which aims to incorporate a much greater degree of automatic reasoning and deep-understanding of web-based content, is largely held to rely on the development of more powerful NLP methods. A principal reason for this is that the metadata structures, such as RDF triples, employed by the Semantic Web are not easy for humans to read or encode. Therefore, the development of this information environment would be facilitated by enabling the Semantic Web to utilize natural

languages (Katz & Lin, 2002). The global proliferation of hand-held computing and communication devices, all of which rely on some kind of natural language-based interface has likewise created an explosion in the demand for more sophisticated NLP technologies. Since the limited size of these devices (Murphy S. , 2012), coupled by the attention-demanding environments in which they are typically used (Przybyla & Zhou, 2008), places a premium on the efficiency of interface design, interactions that require the use of natural language are a prime target for improvement. For instance, Apple's Siri personal assistant program for the iPhone is primarily intended to provide hand-off access to the features available on and the services available through the device.

Lucrative technological and commercial incentives provide a final reason to be interested in the development of NLP applications. A fairly broad analysis of the financially lucrative areas in which NLP technology has already been deployed is provided in (Church & Rau, 1995). A sample of NLP-based businesses reveals how profitable the fruits of this research can be. Nuance, a company that specializes in Speech Recognition Software, posted a 2012 Q3 earning of US $431.7 million (Nuance Communications, Inc., 2012). In 2007, Nuance acquired Tegic Communications, a subdivision of AOL Time Warner that is chiefly known for developing the T9 text prediction algorithm widely used on mobile devices, for approximately US $265 million (Nuance Communications, Inc., 2007). It is noteworthy that the algorithm behind T9 is fairly simplistic in comparison to a standard computer application (United States Patent No. 5818437, 1998). Larger companies have also demonstrated interest in the efforts of NLP-focused start-ups. Microsoft purchased Powerset for about US $100 million in 2007 (Marshall, 2008), Apple purchased Siri for an estimated US $200 million in 2010 (Scoble, 2010), and IBM purchased Language Analysis Systems for an unspecified amount in 2006 (IBM, 2006). Even larger amounts are at stake if the sizes of the global market for specific subfields of NLP are taken into account. Due to the increase in globalization and cross-culture contact that has occurred since the beginning of the Information Age, the demand for language services has massively increased. In 2010, the estimated value of this sector was near US $26 billion (Kelly & Stewart, 2010). While automatic translation services would be unlikely to underlie all of this activity in the near

future, developing them is certainly a priority for the major companies in the language services field.

1.4 Conclusion

The above literature review demonstrates that NLP research is applicable to both Cognitive Science and AI. By better understanding and modeling how the human mind processes language, further insights will be obtained into the cognitive architecture that underlies all of human thought. This knowledge will in turn aid in the development of more effective artificial platforms that will be capable of using human level-language and to reason about real-world problems. There are many incentives for producing such systems, in terms of scientific, philosophical, technological, and commercial benefit. Furthermore, the information compiled in this literature review should serve as ample motivation to pursue future research in NLP technologies using a mixture of Cognitive Science techniques and methodologies. The integrative nature of language truly makes it a prime candidate for multidisciplinary efforts.

Going forward, one thing for the reader to note is that much of the terminology used in this dissertation, such as knowledge-lean and knowledge-rich are loaded terms, but making a distinction between systems along statistical and logical lines is not any more informative. Many texts on Statistical NLP state that poor choices of terminology are endemic to this field and by and large have not been satisfactorily resolved. As an example of further confusion, some of the commercial approaches to NLP that are discussed later on utilize knowledge, but not in a way that tightly integrates it with linguistic parsing or complex reasoning. For instance, a true knowledge-rich parser would use non-linguistic information in an utterance to resolve ambiguous parsing situations; it is known to the author if IBM's Watson handles negation or more grammatically complex sentences in this way. Regardless of whether Watson uses world knowledge to directly inform parsing decisions, clearly Watson is not using knowledge-rich inference if it answers "Toronto" if the question is asking for a city in the United States. Moreover, since Watson learns automatically, it must have some characteristics of statistical approaches in the sense that machine learning algorithms are statistical because they involve inductions based on pattern frequency. It is the view of this

dissertation that to be a knowledge-rich NLP system in the absolute sense, a system must use non-linguistic knowledge to inform parsing decisions and it must also be able to conduct knowledge-rich inference that extends beyond syntactic parsing to handle semantic and pragmatic concerns, as well.

2. NLP Applications

2.1 Introduction

Perhaps because the use of language is such a central human behavior and a major component of human cognition, there are huge numbers of applications that would benefit from the automatic understand of language. As mentioned above, ample motivation for continuing research in the NLP field can be found in the direct economic value of creating some of the more advanced NLP applications, such as a human-level "universal translator" (for an overview of progress made towards this endeavor, see (Shen E. , 2010)). As increasing amounts of information are created and accessed electronically, the market for NLP is steadily growing. However, beyond the monetary worth of NLP technologies, there is also the promise of vast improvements in the delivery of many services and in the progress of science and technology in general. The information technology, biomedical, and legal professions all stand to benefit immensely from advances in automated NLP. Likewise, the effort required to conduct reviews of scientific literature and to obtain existing scientific knowledge would be streamlined. It would also be possible for many documents to be simultaneously published in many of the world's languages. There is also the possibility that there are a number of potential applications for human-level NLP technology that have yet to be popularly identified, but will become more apparent as the technology advances. These benefits all derive from the same underlying capability for machines to process the syntactic, semantic, and pragmatic aspects of natural languages at a deep, human-level of competence.

2.2 Segmentation

Segmentation deals with the establishment of boundaries for meaning-bearing natural language structures, such as words (Van Aken, 2011), sentences (Schmid, 2000), and paragraphs (or topics) (Purver, 2011). Therefore, segmentation is one of the lowest-level NLP sub-problems in terms of the complexity of the content of the linguistic structures that are produced as solutions. The structures in this case are delimiters, namely whitespace and punctuation marks, which themselves hold little semantic content. Segmentation can also be considered a lower-level task from another perspective, since

most other commonly-studied NLP sub-problems require utterances to be segmented into their linguistic units upfront. Despite the apparent simplicity of this problem, a large amount of world knowledge and linguistic context are required for this task, even when proper capitalization and whitespace are present. Consider the complexities of abbreviations, acronym casing, and the structure of embedded equations.

At least for English, segmentation is probably most relevant to the practice of the transcription of text from speech, as there is a great possibility for ambiguity, due to the variation in pauses between speech items and the phonetic structure of words. For instance, there are many multiple word homophones, such as “a rise” and “arise” that are difficult to distinguish in speech without taking context into consideration. However, handwritten, non-standard, corrupted, or error-ridden texts may also require segmentation. Languages, such as Chinese, which do not use alphabets to communicate written information, rely on segmentation processes to greatly reduce the complexity of other NLP tasks. Chinese NLP systems typically must either “preprocess” text with a segmenter or use a joint model that performs both segmentation and the target NLP task simultaneously (Li, Wang, & Yao, 2011). Although these cases are largely beyond the scope of this proposal, which is intended to primarily examine well-formed, written, English-language texts, they are nevertheless important to acknowledge, as they demonstrate that many NLP frameworks, statistical and otherwise, are inadequate to handle the complexities of human-level language use, and are instead restricted to operating in unrealistically constrained and artificial environments.

2.3 Speech and Handwritten Text Recognition

Unlike the processing of digitized written text, the processing of speech and handwritten texts must start by first converting the material into a form that can be directly accessed by a computer. Although most humans, in most circumstances solve these problems with little effort, researchers have yet to create an artificial system that is capable of this type of performance. Moreover, the difficulty of the speech recognition problem greatly increases when there are multiple speakers (Kwan, et al., 2008) or a noisy environment (Gong Y. , 1995). The high demand for automatic speech transcription and recognition has directly led to the use of a large number of algorithms,

which have since been deployed to other NLP-related problems. For example, Hidden Markov Models, which will be described in a future section of this dissertation, were utilized as early as 1975 (Jelinek, 1976) to solve the speech recognition problem using statistical methods. Similar types of statistical models (Taskar, Guestrin, & Koller, 2003) can be used for optical character recognition, which may be used for transcription from handwritten or analog texts into a digital medium. The Hidden Markov framework eventually became one of the best understood and most developed of the frameworks for statistical NLP. However, achieving human-level accuracy for speech recognition, even in ideal environments, has proved elusive. Even such a relatively simple and isolatable NLP task as the recognition problem actually requires a great deal of sophistication and knowledge to solve at the human level. Unlike most models for speech and written character recognition, the human cognitive system appears to leverage aspects of many different aspects of linguistic processing. Indeed, neurological studies have indicated that there is a high amount of top-down influence in human speech processing, even at the perceptual level (Garnes & Bond, 1976).

2.4 Text Completion and Spelling and Grammar Correction

There is a great demand for technologies that facilitate the entry of text into electronic devices. Due to the vast amount of textual information that is now created on some kind of electronic platform, it is clear that a great deal of time and effort would be saved if the process of creating such text was streamlined. Moreover, there is an ever-expanding usage of mobile devices and other systems that have interfaces that are not optimal for the entry of text (MacKenzie & Soukoreff, 2002). There are many situations in which someone may want to send a text message, but has limited time or physical resources (e.g. while driving) to do so. In addition, some types of disabilities force humans to communicate through body driven interfaces, which, without advanced techniques to streamline the input of language, drastically reduce the speed of communication.

The idea of manipulating the efficiency with which machine-generated text can be created has existed at least since the development of the QUERTY keyboard

configuration for typewriters (United States of America Patent No. 207559, 1878)[4]. One of the currently most widely used NLP technologies, T9 text completion (United States Patent No. 5818437, 1998), was created to facilitating text entry on mobile devices. However, despite the popularity and utility of this application, the T9 program is extremely limited in its linguistic complexity, which is clear from an analysis of how it decides between "collisions" in world selection. On a standard phone keypad, there are a number of collisions between very common words, wherein multiple words could be spelled with the same sequence of key presses. Usually, it would be obvious to a human which word was intended based on immediate context alone. The T9 algorithm fails to take this source of information into account, leading to such predictions as "do you know **he** your flight is on time" instead of the more sensible "do you know **if** your flight is on time." Foreign languages pose unique difficulties for text suggestion technology (MacKenzie & Tanaka-Ishii, 2007).

Text completion is also useful even if a full keyboard is available. For instance, some search engines, such as Google (Google, 2013), store many of the most common search queries and high-frequency associations between search terms. By curating this information, it is possible for suggestions for competing partial search queries can be made available for users in real time. However, there are limitations to this technology. Unusual searches may be automatically modified to more common queries that do result in retrievals that are related to the user's original intention. Also, suggestions may not be appropriate for certain uses: Google was sued by a Japanese man whose name was associated with "autocompleted" searches that involved criminal activity (BBC News, 2012). This type of problem results in part due to the lack of any inferential capability or structured knowledge in the Google system.

The inconvenience caused by an over-reliance on knowledge-lean, shallow NLP technologies extends to the domain of word processing. One of the seemingly simplest and most frequently encountered NLP applications is spelling correction. However, there are many pairs of words that are either similarly spelled (e.g. "step" and "steep") or homonymous (e.g. "vane" and "vein"). Even knowing a word's the part of speech is not enough to disambiguate a sizable number of these cases in a minimally inflected

[4] Note though that the QUERTY keyboard was designed to *reduce*, not increase typing efficiency.

language, such as English. This situation occurs because identical open-class words can often serve in more than one grammatical role (e.g. “step” and “steep” can both infinitive verbs; “vane” and “vein” can both be nouns). In the Brown Corpus (Francis & Kucera, 1964), a hand-tagged compilation of approximately 1,000,000 words worth of English text, 48% of the words that appear can be mapped to more than one part of speech (DeRose, 1992). Although statistical modeling is partially capable of modeling part of speech tagging using nearby words alone, the full grammatical context of a word is required for improved accuracy. Modern techniques for spelling correction of search queries are capable of between 15% and 49% accuracy depending on the type of error (Li, Duan, & Zhai, 2012). Grammar correction is an even more difficult problem (Wagner, 2012). Most commercial grammar checkers are based on shallow parsing techniques that make use of a very limited number of rules. It is necessary to go far beyond this level of complexity to approach human-level performance (Clement, Gerdes, & Marlet, 2011).

However, it will also be demonstrated that systems that actually do ostensibly make use of knowledge-rich representations and learn from knowledge-rich data sources are not necessarily any better suited for achieving human-level NLP than knowledge-lean systems. It turns out that, by some kind of disconnect between groups in academic and commercial research into NLP, academic systems tend to favor knowledge-lean, statistical approaches, while a number of recent high-profile commercial systems are based largely on knowledge-rich methods. Knowledge-rich techniques account for at least some component of the architecture of Wolfram|Alpha and Siri; IBM’s Watson project is based almost entirely on knowledge-rich methods. Recent research presented by Naveen Sundar at Rensselaer Polytechnic Institute (Sundar N. , 2013) (Sundar, Bringsjord, & Licato, 2013) sheds some light on the mechanisms at work in the Watson system and indicates that a major arm of future research involving Watson will work towards extending its capabilities as a knowledge-rich system. Nevertheless, it is one of the main thesis of this dissertation that it is necessary for several key characteristics to be exhibited by any system, whether knowledge-lean or knowledge-rich, that aims towards human-level NLP. Namely, such systems must be able to seamlessly reason about linguistic and non-linguistic representations efficiently in the same framework; such

systems must also involve self-adapting algorithms to provide flexible inference techniques in order to handle the extreme complexities associated with human-level inference problems; finally such systems must also be able to seamlessly integrate special-purpose algorithms and representations with general purpose ones—it will be shown to be possible to map most human-level problems onto a fairly small collection of specific domains, each of which can be made much more efficient by implementing special-purpose computation machinery. It is not clear that any of the state-of-the-art commercial systems are designed to take these concerns into account. Therefore, this provides an explanation for why a system that was designed to accommodate these concerns is capable of outperforming vastly more expensive and extensive commercial systems on a number of tasks. Empirical evidence of this claim will be presented near the end of this dissertation.

2.5 The Semantic Web and Natural Language Search Engines

There are several factors that have led to the rapid growth of information in human culture. Some of the principal causes for this are the immense popularity of the Internet as a medium for information transfer and the wide-spread availability of machine-assisted content creation. The data accessible from the Internet provide an incalculable benefit for a vast number of human behaviors, from art and communication, to science and politics. According to the founders of the Sematic Web initiative, the Semantic Web is primarily intended to "bring structure to the meaningful content of Web pages" (Berners-Lee, Hendler, & Lassila, 2001). Yet, the presence of structure does not entail the full functionality that is envisioned for this cyber-institution. It is clear that some amount of sophisticated inference must be provided that can operate on the web of data. Such inferential tools, notably the RACER engine for deriving inferences from OWL ontologies (Protege, 2009), have an expressive power limited to that of description logics. This restriction reduces the expressivity of statements that are expressible with these languages, which serves as a tradeoff with the apparent intractability of uncontrolled natural languages (Schwitter & Tilbrook, 2004).

Moreover, the semantic web's requirement of a high volume of accurate metadata has led some researchers to question whether it is possible to create this type of network

without relying on some kind of automatic NLP technology (Wilks & Brewster, 2009). Even if it is the case that humans can be trusted to provide the annotations that provide the backbone of the semantic web, the RDF format that is currently described in the WC3 Standards for the semantic web (W3C, 2013) is not a natural way for humans to express world knowledge (Katz, Lin, & Quan, 2002). Whether or not sophisticated NLP is a strict requirement for the semantic web, empirical studies have revealed that even novice users benefit from a natural language interface for interacting with knowledge bases that are derived from ontologies (Kaufmann & Bernstein, 2007).

2.6 Biomedical and Legal Documents

Although the problem of domain general natural language search overlaps with some of the aspects of mining information from biomedical and legal documents, the particular demands generated by the professionals of those fields for improved search tools has led to a large amount of work specifically tailored to these areas. A large amount of world knowledge is required for any robust understanding of much of the text produced by the biomedical and legal professions (Nirenburg, McShane, Zabludowski, Beale, & Pfeifer, 2005). Moreover, achieving a very high degree of accuracy is extremely important in processing these documents. As with many NLP applications, there are a number of sub-problems associated with the understanding of professional texts that are valuable to solve in their own right. For instance, medical data is often recorded with idiosyncratic shorthand and terminology, which may be unique to an individual practitioner (Mystre, Savova, Kipper-Schuler, & Hurdle, 2008).

Three broad classes of biomedical documents have been identified by researchers in this domain, each of which poses specific problems for NLP systems (Patrick, 2010). First, medical papers are the most conforming to general NLP techniques, but contain a large amount of complex domain-specific terminology. Second, formal clinical reports contain even more jargon and complex listings of quantitative information (e.g. weights or dosages). Third, clinical notes may contain information from multiple languages, non-standard grammatical usage, idiosyncratic terminology and abbreviations, as well as a significant number of errors. An extensive amount of knowledge about specific medical practices may not be necessary for a first pass solution to disambiguating confusing

terminology, but the greater the demand for accuracy, the greater amount of knowledge that is required. Even though a task like Named Entity Resolution is on the less complex side of the NLP spectrum, this has proved to be an essential technology for the automatic processing of texts in the biomedical domain (Settles, 2004). Therefore, even though human-level NLP requires a great deal of general, commonsense world knowledge, the need to understand documents with technical content creates a demand for even more complex modeling techniques.

Legal documents contain many of the same difficult-to-parse language-related phenomena as biomedical documents. Likewise, there is a wide range of inferential tasks that must be performed in the course of generating and understanding legal documents. Many legal documents contain references to laws that can be expressed as formal logical rules. Inducing these rules from natural language texts is a non-trivial NLP task, which may be facilitated by the utilization of semantic models (Francesconi, 2010). Checking legal matter for consistency is another area that requires sophisticated inference (Dinesh, Lee, & Webber, 2006). Counterfactual reasoning also figures in legal documents. Hypothetical situations (as opposed to actual court cases) are often anticipated in legal decisions, because court cases frame guidelines (e.g. legal precident) for the future application of the relevant laws. This practice is a type of example-based reasoning that is common to other areas of cognitive science and AI research (Rissland, 1983).

2.7 Education

Standardized testing and a number of other types of academic assessments require NLP capabilities beyond the need to automatically detect mechanical and grammatical errors in text. One area of great interest to educators is long-response evaluation. The automatic evaluation of essays and other long-response assessments is one of the most difficult, variable, and time-consuming tasks faced by instructors. Many standardized tests at the primary, secondary, and post-secondary levels contain at least one long-response question. Additionally, courses in many subjects at every level of education require at least some amount of structured writing. It has been found that increasing secondary students' exposure to essay writing may result in significant improvements in their total academic competency (Tyre, 2013). Easing the burden of grading these types

of assignments would allow teachers the opportunity to increase their frequency in the standard curriculum. Furthermore, grading qualitative test answers is a highly variable process when performed by humans; inter- and intra-grader scores are known to be unreliable under certain conditions (Brown, 2009). Automatic graders have been shown to ameliorate some of the unreliability and validity problems associated with essay-based assessments (Wang & Brown, 2007).

On the other hand, NLP technology may even be used to actually generate the content of assessments. The ability to produce valid questions for some arbitrary material on the fly would be valued not only by individual instructors, but also large testing agencies, such as the New York State Board of Regents or ETS. There are a number of technically-involved issues (beyond the scope of this dissertation) involved with this task, which are currently being explored from a number of perspectives (Gierl & Haladyna, 2013).

2.8 Other Applications

The demand for NLP technologies is both vast and diverse. While many target applications for NLP do not necessarily require full human-level capability, most are complex enough to surpass the capabilities provided by the current state-of-the-art, and likely will continue to remain open problems for years to come. However, it is also the case that almost every application for NLP can be scaled down to a level compatible with present technology. This enables tangible progress to be made and material rewards to be reaped from relatively short-term projects—these same points are true of AI in general—if no progress in the field could be made without a complete solution, then the task would be impossible. It is likewise important to continue to push towards increasingly sophisticated systems rather than to continuously reduce expectations to what can be handled at this time.

Some examples of additional AI applications are: social research (Li & Larsen, 2011), digital library management (Chowdhury, 2010), computer programming (Miller L. A., 1981), automatic translation, multilingual information retrieval (Oard, 1998), question-answering systems (Quarteroni & Manandhar, 2009), improved human-

computer interfaces (Maybury & Wahlster, 1998), geriatric care (Augusto & Nugent, 2004), and molecular modeling in chemistry (Weininger, 1988).

3. NLP Subproblems

3.1 Introduction

Language is clearly a highly integrative cognitive function, both in terms of its links to other domains of cognition, such as perception, and in terms of the various faculties that span the different levels of granularity of linguistic processing. For instance, processes such as phonological perception would typically be considered a low-level function, while maintaining mental models of discourse (Johnson-Laird & Garnham, 1980) would be a high-level one. The specific application areas enumerated in the section above involve the full spectrum of human-level language processing. As would be expected, the lower-level cognitive processes feedforward up the cognitive process hierarchy and thus influence the actions of higher-level functions. However, it is also the case that lower-level processes are impacted by feedback from higher-level ones (Stone, Vanhoy, & Van Orden, 1997). Since the algorithms that are implemented by actual neural circuitry are as of yet not completely known, NLP systems (especially of the knowledge-lean variety) are usually in line with one of two general approaches to modeling (Breiman, 2001). The first approach is to use a statistical model to learn patterns in data according to the assumptions implicit in that model. The second approach is to learn the function (or mechanism) through which the data are generated.

Both of these approaches make fundamental assumptions about the nature of the systems that generate natural language data (i.e. human minds) that are probably unfounded due to the lack of knowledge about the operations of the underlying mechanics of these language generation systems. Most NLP projects focus on a narrow portion of the variance in language data in isolation from other aspects of linguistic and non-linguistic cognition. These sub-problems are also often abstracted away from actual cognitive processes, so there is often no guarantee that they correspond to some actual function in human language understanding, nor is there an attempt usually made to attempt to establish this correspondence. Attempting to generate a truly comprehensive view of the structure of the NLP field is somewhat problematic due to the diversity of terminology and the sheer amount of literature produced on the subject. Therefore, this document will defer to an external outside source of expertise on this matter. At the time

of the creation of this document, the ACLWiki[5] (Association for Computational Linguistics, 2012) maintains a compilation of the state-of-the-art systems for a large number of NLP problems and provides a links to a number of resources that are devoted to the NLP and Computational Linguistics field. Many of the references for the state-of-the-art NLP techniques below were obtained from the ACLWiki site.

3.2 Part of Speech (PoS) Tagging

Part of Speech (PoS) Tagging is the task of designating the morphosyntactic role lexical items play in constituting the meaning of the larger structures of which they are a part. This problem statement relies on the assumption that there are a fixed number of categories to which all lexical items can be assigned. The idea that for any given language there exists a limited and universal set of parts of speech has been accepted since ancient times, with proponents reaching as far back as speakers of classical Sanskrit (Matilal, 1990) and Greek (Kemp, 1986). Whereas these prototype theories are much simpler than their counterparts for most modern languages, the notion that parts of speech are as real a component of natural languages as words and sentences has persisted over time. In English, there are typically thought to be eight basic types of PoS designations, which are familiar to most people who have studied the mechanics of the language: noun, verb, adjective, adverb, preposition, pronoun, conjunction, and interjection. Even though the classification of words by these broad distinctions is a useful pre-processing step for many other NLP problems, researchers have found it to be more useful to subdivide these high-level morphosyntactic categories.

One of the most influential of these "tag-sets" is used in the Penn Treebank corpus (Marcus, Marcinkiewicz, & Santorini, 1993). Under the Penn Treebank system, there are currently 36 tags (plus 12 tags for common punctuation symbols), each of which are assigned to words according to a rigorous set of guidelines (Santorini, 1990). An example of a Penn Treebank PoS tag that is more specific than one of the traditional eight tags is: NNS, which designates a noun that is in plural form. With this level of precision, it is easier to generate more in-depth parses of natural language utterances, because knowing both the general part of speech and details such as number, tense, or degree

[5] ACL is an acronym for Association for Computational Linguistics

strongly constrains the possible roles that a word can play in a sentence and contributes to establishing the relationships between words through demands on agreement (e.g. a plural noun can only be the subject of a plural verb). The best statistical NLP approaches attain approximately 97% overall accuracy (with approximately 90% accuracy for words that were not encountered during the training phase) (Toutanova, Klein, Manning, & Singe, 2003); rule-based approaches can perform slightly better, with 98.5% accuracy (Tapanainen & Voutilainen, 1994),[6] but are rarely used because they are inflexible when presented with novel situations. Because PoS tagging can be solved with such a high degree of accuracy, like segmentation, it is typically assumed to be a source of preprocessed information automatically available to platforms for more involved types of NLP tasks—the practices that follow from such assumptions will be shown to have serious repercussions for the field in Chapter 8, which analyzes the evaluation metrics favored by many NLP researchers: PoS tagging is not as accurate as it seems.

Despite the facts that detailed rules for applying PoS tags have been compiled and that state-of-the-art approaches are effective at producing such taggings automatically, efforts are still being made to improve this technology. Like all aspects of natural language, there are large amounts of exceptional and idiosyncratic cases that cause the explicit enumeration of PoS tagging rules to be intractable. Furthermore, in some cases, relying solely on intra-sentence cues is not sufficient to disambiguate a particular PoS tag. A well-known example (Ludeling & Kyto, 2008) is: *The duchess was entertaining last night*. Here, *entertaining* can be cast as either an adjective or a verb participle, depending on the interpretation of the sentence. While the Penn Treebank allows for words to be tagged with more than one part of speech, whoever creates an utterance is almost certain to have one interpretation in mind (puns and other figures of speech notwithstanding). It may be impossible, as in the example above to recover which of the possible interpretations was intended based on the context of a single sentence. Most PoS taggers are incapable of incorporating information beyond the scope of the current sentence that is being tagged.

If disambiguating evidence happens to be available, that information will more likely be couched in the pragmatic, discourse-level of communication, something that

[6] The accuracy for unknown words was not given in this publication.

typical PoS taggers do not consider. Most NLP systems are scoped to examine sentences in isolation, whereas humans retain a memory of aspects of the entire discourse. Furthermore, ambiguities do not "break" human parsers and generally resolve over time; PoS taggers are almost universally unable to adapt during processing. Although humans are thought not to perform tasks such as PoS analysis separately from other modes of linguistic processing (Ball, Freiman, Rodgers, & Myers, 2010), PoS tagging is often studied as a standalone problem. These discrepancies between NLP systems and human language processing may explain some of the discrepancy between machine-level performance and human-level performance in this area.

One of the methods that can be considered as the current state-of-the-art method for PoS Tagging (Denis & Sagot, 2009) obtains an F-score of 97.80% accuracy. According to (Manning C. , 2011), computational researchers have estimated that inter-human reliability in PoS tagging is limited to about 97%, thus suggesting that PoS taggers are, by some measures, near human-level in performance[7].

3.3 Shallow Syntactic Parsing or Chunking

One of the principal properties of natural languages is that they are organized to transmit information using a hierarchy of structures. At the top level, there are documents, paragraphs, and sentences. Beneath sentences there are phrases, which can themselves be comprised of some combination of smaller phrases and individual words. For most NLP tasks, it is sufficient to consider words to be on the bottom layer, but they can be decomposed further into morphemes, which are made up of phonemes and even finer levels of detail after that. The human cognitive system is adept at extracting the hierarchical structure from many different kinds of input; however, this is a process that is difficult to program declaratively. Because the problem of establishing the full network of relationships between the components of language is so difficult, some researchers have tried to solve the seemingly less difficult problem of producing partial, shallow parses of utterances. Typically, this process involves chunking together adjacent words in an utterance according to the functional boundaries of their parent phrases. For

[7] See the section on NLP Metrics for a more in depth analysis.

instance, in the sentence *The dog is barking*, two chunks might be identified: *The dog* would be designated as a noun phrase; *is barking* would be designated as a verb phrase.

Once this chunking stage is complete, sentences are rendered into their principal components, which generally facilitates the identification of the higher-level relationships (e.g. *the dog* is the subject of *is barking*) necessary to extract semantic information. Aside from its favorable computational complexity, chunking appears to have a greater degree of cognitive plausibility than exhaustive syntactic parsing (Brooks & Center, 2003). Further information about the hypothetical role that chunking plays in human sentence processing can be found in: (Abney, 1995).

The current state-of-the-art method for Noun-Phrase Chunking (Shen & Sarkar, 2005) obtains an F-score of 95.23% accuracy.

3.4 Prepositional Phrase Attachment

One of the major difficulties faced by researchers in the NLP field is how to handle the vast amount of ambiguity that is present in natural language use. Ambiguity is usually thought of as a product of some instance of a semantic or syntactic structure that is open to multiple plausible interpretations. Syntactic or structural ambiguity is one of the major problems encountered by full syntactic parsing. Even though chunking techniques, as described above, attempt to avoid the costs of more complete parsing, it is still necessary for many NLP applications to determine something about how the phrases of a sentence fit together. In English, one of the most commonly encountered ambiguities in phrase attachment is whether a prepositional phrase attaches to a verb (e.g. it describes the manner in which something is done) or to a noun (e.g. it describes a characteristic of an object).

For example, in the sentence *I saw the man with the telescope*, the prepositional phrase could attach to either the verb *saw* or the noun *man*. If the former attachment is done, the sentence means *I used a telescope to see a man*. If the later attachment is done, the sentence means *I saw a man who had a telescope with him*. Since the different attachments of the propositional phrase lead to somewhat different interpretations, it is clear that an NLP system used for military or medical purposes would need to make the correct decision. Note that when a prepositional phrase attaches to a noun, it can help to

disambiguate the intended referent. In a combat situation where there are multiple possible targets, telling a drone to *Shoot the man with a missile* could lead to unwanted results.

There are many sources of potentially disambiguating information to help solve the phrase attachment problem, but not all of them are linguistic in nature. To illustrate why non-linguistic world knowledge might be needed to determine the proper attachment, consider the following sentence that is similar in structure to the previous example: *I saw the star with the telescope*. Although the argument could be made that *with the telescope* could attach to either *saw* or *star*, most humans would find it highly unlikely that a *star* would be in possession of an object. The only way to determine this is to have some operational knowledge about what a star is and what types of entities are likely to possess an object.

The state-of-the-art accuracy for prepositional phrase attachment is between 84% and 85%. Two approaches that use different knowledge-lean machine learning approaches were able to obtain this level of performance: a backed-off n-gram model (Collins & Brooks, 1995) and a maximum-entropy-based model (Ratnaparkhi & Reynar, 1994). Human performance appears to have a maximum accuracy of 92.3%, according to (Ratnaparkhi & Reynar, 1994).

3.5 Deep Syntactic Parsing

The production of deep syntactic parses of natural language utterances (usually sentences) is one of the most studied NLP tasks (Klein & Manning, 2003). However, there is some experimental evidence that suggests that humans may not actually produce such parses when they process language (Sporleder & Lapata, 2005). Generating a parse of a sentence involves recovering the "deep structure" of information that is encoded and communicated through linear speech or text. Unlike chunking, deep parsing requires the identification of all of the relationships between all of the words and phrases in a sentence. Natural languages have recursive grammar rules, so phrases can be embedded in larger phrases of different types, all the way to the sentence-level. As sentences increase in length, the number of possible parses to consider increases by a large factor.

Because there are many parse structures for every sentence, not all of which are necessarily grammatical, parsing is a computationally complex problem.

Due to the fact that natural language utterances contain recursive structures, they can only be captured by representational formalisms of sufficient expressive power. It is possible to order types of formal languages in a strict ordering based on their complexity. This ordering is usually referred to as the Chomsky Hierarchy, as it was most popularly identified by linguist Noam Chomsky (Chomsky, 1956). Chomsky also famously illustrated that languages that admit recursion are at least as complex as a class of formal languages known as Context Free Grammars (or CFGs) (Chomsky, 2002). This implies that modeling techniques that are not at least as powerful as a CFG parser would not be able to process natural language at a human-level of performance.

Like natural languages, formal languages can be parsed; however, there is a major difference: unlike natural languages, formal languages can be described with a fixed set of precise rules. The complexity of the structure of those rules determines the complexity of the language. For instance, CFG rules must be of the form $N \rightarrow x$, where N is a "non-terminal" symbol and x is a string that can consist of combinations of terminal and non-terminal symbols. In the context of CFGs, terminal symbols correspond to the elements of the language's "alphabet," which in natural languages would usually correspond to words or other lexical items. On the other hand, non-terminal symbols are abstract and serve to provide structure by grouping together terminal and other non-terminal symbols in a hierarchy (these are analogous to natural language phrases). Because a CFG rule can contain a non-terminal symbol in its right hand side, recursion is possible through rule sets such as: $\{A \rightarrow aa, A \rightarrow bBb, B \rightarrow A\}$. For a natural language, formal language-like rules might serve to describe a subset of its possible constructions, but, inevitably, idiosyncratic usages make it infeasible to characterize natural languages with a fixed set of exact rules. To handle the imprecision of natural language grammars, a non-deterministic extension of CFGs, known as Probabilistic Context-Free Grammars (PCFGs), was identified. PCFGs permit a level of flexibility because their rules are not absolute, but rather are associated with rule weights that are somehow related to an estimation of the probability that the rules hold over all utterances in the language.

Although there are also tractable methods for PCFG parsing (Stolcke, 1995), it turns out that further investigations into the phenomena of natural languages have determined that some natural languages are more complex than CFGs: they are at least as complex as a class of languages known as Context-Sensitive Languages (CSLs) (Shieber, 1985). Parsing the general class of CSLs requires the equivalent of a PSPACE complete algorithm (Kuroda, 1964), which would, given the present state of technology, require more computational resources than is practical. Some theories of natural language grammar, such as Head-driven Phrase Structure Grammar (HPSG) (Pollard & A., 1994), belong to the class of recursively enumerable languages, which are even more complex than CSLs (Carpenter, 1991). Accordingly, new research has been done to identify tractable parsing techniques for languages that are between CFGs and CSLs (e.g. Mildly Context-Sensitive Languages (Joshi, Vijay-Shanker, & Weir, 1991)). Nevertheless, the current state-of-the-art parsers primarily rely on modeling natural languages as PCFGs.

Aside from the computational complexity required to parse the class of formal languages that incorporate the features of natural languages, there are additional obstacles to creating an efficient human-level NLP parser. In natural language utterances, many liberties can be taken with the formal grammar that supposedly defines the language in which it is uttered. This degree of flexibility often causes difficulties for parsing algorithms, even if they are based on stochastic processes. For instance, the phenomenon of ellipsis (Lappin, 1995) is an aspect of natural speech that greatly confounds the problem of producing parses. Ellipsis removes one or more words from a sentence, which are patched in by humans using world-knowledge and pattern completion. A common example of ellipsis occurs with conjunctions in which the first conjunct is truncated. The sentence *John drove and Mary walked to the store* would normally be understood by a person to mean by that *John drove to the store and Mary walked to the store*. However, there are similar grammatical constructions in which no ellipsis would be detected by a human judge. Consider the example *John slept and Mary walked to the store*. Slept is an intransitive verb, so it is unlikely that a human would interpret this sentence to mean *John slept to the store and Mary walked to the store*. The existence of such null cases makes the detection of an ellipsis a difficult problem in itself. In order to

cope with the influence of individual words on the resultant phrase structure, "lexicalized" methods for parsing (Resnik, 1992), which take the identity of the main words in an utterance into account for processing purposes, were introduced to improve accuracy.

There are two common types of parsing outcomes: 1.) a constituent parse or 2.) a dependency (or link) parse. A constituent parse results in the full phrase structure of a sentence, in which all words are grouped into leaf constituent phrases. These leaf constituents are then grouped into non-leaf constituent super-phrases that bear some resemblance to the X-bar phrases of Chomskyan parsing theory (Chomsky, 1970). The ultimate node of a constituent parse is a sentence-level node that spans the entire sentence. Figure 1 depicts a graphical representation of a constituent parse for the sentence *This is a test sentence* by an online parser (Carnegie Mellon University, n.d.), which is used in all of the examples mentioned in this paragraph. Figure 2 depicts how the constituent structure of this sentence would be expressed in a corpus. This style of annotation is known as bracketing, in which the phrase and word contents of the brackets represent the relevant structure of the sentence.

A constituent parser aims to replicate the same brackets as would appear in the human-generated gold standard. Bracketing annotations are used to train NLP systems and also to evaluate how well those algorithms performed.

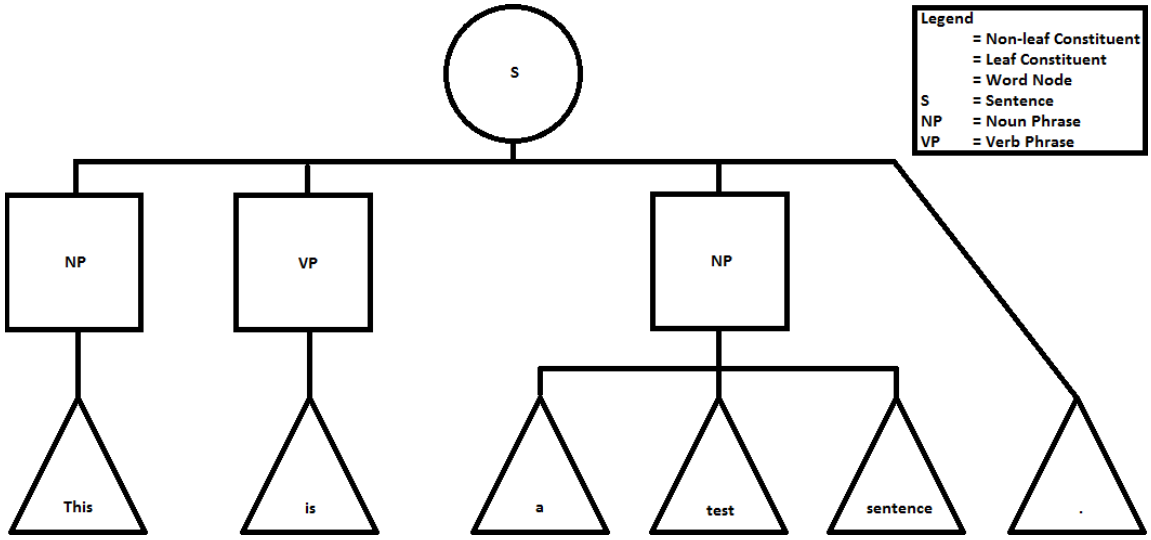

Figure 1. A graphical representation of a constituent-type sentence parse.

```
(S (NP This)
   (VP is
       (NP a test sentence))
   .)
```

Figure 2. How constituents are labeled in the Penn Treebank corpus (Penniston & Harley, 2011). Note that the formatting of the labeling is intentional.

On the other hand, a dependency parser determines the functional relationships between leaf-level phrases rather than identifying the entire phrase tree structure. Therefore, such methods may make use of shallow parsing methods to identify the leaf phrases and then use additional methods to determine how those phrases are linked together. Dependency parsing can be useful for extracting some semantic information from a sentence even if the overall parse is not entirely correct. A depiction of the result of a dependency parse for *This is a test sentence* can be found in Figure 3.

```
    +------------------Xp-----------------+
    |               +-------Ost------+     |
    |               |  +------Ds-----+     |
    +---Wd---+-Ss*b+  |    +---AN---+     |
    |        |     |  |    |        |     |
LEFT-WALL this.p is.v a test.n sentence.n .
```

Figure 3. A graphical representation of a dependency-type sentence parse (Yli-Jyrä & A., 2005)

In most languages, including English, constituent parses and dependency parses generally result in approximately the same accuracy, but constituent parsing has received more attention from NLP researchers (Nivre, 2005). However, for some languages, such as German, which have a high frequency of long-distance clausal relationships, dependency parses may be more accurate (Kubler & Prokic, 2006). Other languages, such as Chinese, might benefit from hybrid parsing methods that combine features of constituent and dependency parses (Wang & Zhang, 2010).

The current state-of-the-art method for Deep Constituent Parsing (McClosky, Charniak, & Johnson, 2006) obtains an F-score of 92.1% accuracy. For Deep Dependency Parsing (Sleator & Temperley, 1995), the state-of-the-art F-score is 86.3%. Both of these systems were evaluated on a version of the Penn Treebank corpus. When human experts are instructed to produce a constituent phrase parse of a corpus, they show approximately a 94% correspondence (Sampson & Babarczy, 2006). This suggests that this level of accuracy is about the highest that can be expected from an NLP system. Moreover, these results have led researchers to be optimistic that PCFG-parsers are sufficient to produce high quality parses of natural languages, even if natural languages actually belong to a different formal complexity class.

3.6 Word Sense Disambiguation

Natural languages exhibit a high degree of ambiguity across all levels of discourse. In some languages, such as English, this problem is exacerbated due to the fact that many English words have multiple senses of meaning. Sometimes, these senses are polysemous (i.e. semantically related), as in *school* as a place of learning and *school* as the set of people who attend a particular school. However, due to the phenomenon of homonymy, the same word can have multiple distinct meanings, as in *bank* as the landform surrounding a river and *bank* as a place in which something can be secured. From the above examples, it is clear that even if the part of speech of a word is known, there is still a great deal of ambiguity left to process. In order to retrieve the best fitting sense of a word, some information about the word's context must be used. Understanding context of a word not only subsumes the ability to produce at least a shallow parse of the sentence of which it is apart, but also may require information about

other portions of the discourse, as well as a large amount of world knowledge. The complexity of this task has led to the development of many different kinds of approaches for solving this problem, some of which use more knowledge-rich approaches and others that rely on statistical processing (Stevenson & Wilks, 2003). To augment the use of direct-from-text contextual information, some researchers utilize WordNet (Leacock & Chodorow, 1998), which allows NLP systems to access precompiled information about the different relationships between word senses.

The current highest performing method for Word Sense Disambiguation (Yarowsky, 2000), which relies on the automatic extraction of boilerplate features to identify the most likely sense of a word, obtains an F-score of 69% accuracy. Although it is difficult to compare the performance of machine learning techniques that are evaluated in different ways from human performance, experiments that involve the selection of word senses by humans provide an idea about how difficult the Word Sense Disambiguation problem is. A psycholinguistics study (Ahlswede & Lotand, 1993) demonstrated that human judges average only a 66% agreement rate on determining the sense of words from the same test passage. Thus, it may be the case that the level of naturally unresolvable ambiguity inherent in this task would limit the performance of any system designed to solve it.

3.7 Co-Reference and Anaphora Resolution

Understanding a document (or a series of related documents) with multiple instances of named entities is complicated by the need to establish the correct identity relationships between each occurrence. Even if a document only contains multiple uses of the same name, identity can be a difficult issue due to corner cases. For instance, two identical names in a text sometimes refer to distinct individuals (e.g. *This is my brother Daryl and my other brother Daryl*). Another confusing case involves what is known as use vs. mention, in which an expression is meant to refer only to the name of an entity and not the entity itself (e.g. Wikipedia article for the name *John Smith* (Wikipedia, 2013)). References can also involve a mixture of pronouns (e.g. *he*) or generic type nouns (e.g. *the man*) and named entities (e.g. *John Smith*). Matters are further complicated by the fact that the generic mention can occur either in after the initial

mention of the specific entity, as in anaphora (e.g. *John took a nap before he went to the store*), or before the initial mention of the specific entity, as in cataphora (e.g. *Before he went to the store, John took a nap*). Resolving names with epithets (e.g. *Barak Obama, President of the United States*) or abbreviated (e.g. *Barak Obama…Obama*) or alternative (e.g. *Barak Obama…Barry Obama*) forms is an additional source of difficulty.

The use of world knowledge is crucial to surmounting these difficulties, as speakers often assume that their audience has the same background knowledge (e.g. who is the current President of the United States of America) and common sense knowledge (e.g. that people can be referred to by either their first or last names). Even more complex cases involving knowledge-based reasoning are discussed in detail in: (McShane, 2009). Despite this reliance on knowledge-rich inference, like most current NLP projects, many co-reference resolution systems utilize a knowledge-lean statistical approach. In order to accommodate the complexity of this task, NLP researchers typically restrict the cases that they handle to the simplest possible varieties. An examination of the guidelines for the MUC-7 conference, which focused, in part, on co-reference resolution, reveals the limitations on the types of expressions present in its evaluation corpora (Hirschman, 1997).

The current state-of-the-art method for Co-Reference Resolution (Stoyanov et al., 2010) obtains an F-score of 56-80% accuracy, depending on its threshold settings and the data set used for evaluation.

3.8 Other Sub-Problems

Because of the sheer number of approaches and due to the fact that different researchers use different terminology to describe similar tasks, use the same terminology to describe different tasks, and delineate sub-problems along different lines, it is essentially impossible to provide an exhaustive account of every NLP task that might be encountered in the literature. A selected list of various problems and associated resources follows (c.f. (OpenCog, 2012) (Wikipedia, 2013)), but is by no means meant to be exhaustive: textual entailment (Dagan, Glickman, & Magnini, 2006), sentiment analysis (Pang & Lee, 2008), text classification (Sebastiani, 2002), discourse analysis (Soderland & Lehnert, 1994), stemming (Frakes, 1992), natural language generation (Reiter & Dale,

1997), semantic role labeling (Palmer, Gildea, & Xue, 2010), relationship extraction (Giles & Wren, 2008), named entity extraction (Nadeau & Sekine, 2007), information retrieval (Lewis & Jones, 2996), text alignment (Veronis & Langlais, 2000), question answering (Voorhees, 1999), information extraction (Cowie & Lehnert, 1996), automatic summarization (Nenkova & McKeown, 2011), and truecasing[8] (Lita, Ittycheriah, Rouk, & Kambhatla, 2003).

[8] Truecasing is the practice of restoring the text casing in documents in which such information has been damaged.

4. NLP System Designs

There are several broad categories into which NLP systems may be classified. One way to categorize the kinds of systems that most often appear in the field is to split them between the following three categories: 1.) isolated task systems, 2.) pipelined approaches, and 3.) unified approaches. In general, these types of systems can be ordered in increasing complexity and generality according to the sequence in which they appear below.

4.1 Isolated Task Systems

Most NLP projects have involved applying a particular formalism to a particular NLP sub-problem, such as any of the individual problems described above. Such narrowly-scoped systems are generally capable of solving only fairly low-level NLP tasks and require a large amount of preprocessing to work in the first place. For instance, a stand-alone parser assumes that at least the segmentation and PoS tagging tasks can be solved perfectly. These are bad assumptions, since the error rate for these tasks is significant in practice and faulty pre-processing almost always results in an incorrect final analysis. It is unlikely that any such programs would be able to attain human-level performance on the task of general understanding of utterances. However, in fairness, most of these projects are not intended to do more than provide empirical data on how well particular machine learning algorithms can handle linguistic problems. Although such efforts are useful for certain applications, it is certainly necessary to expand beyond their limitations if the goal of human-level NLP is ever to be reached.

4.2 Pipelined Approaches

In order to approach a human-level understanding of natural language utterances, it is almost always necessary to combine the efforts of several NLP task-solvers. The traditional way to accomplish this is to array individual programs into a purely feed forward pattern. For instance, a pipeline might consist of the following components. Stage 1 would be a segmenter to break text into words, sentences, and paragraphs. Stage 2 would be a PoS tagger. Stage 3 would be a Word Sense Disambiguation system. Stage 4 would be a shallow parser. Stage 5 would be a deep parser. Stage 6 would be a

reference resolution system. Stage 7 would be an information extraction routine. While pipelines are more sophisticated, and in a sense more realistic, than isolated task solvers, they do not exhibit on of the most crucial properties of the human cognitive system: that top-down processing is at least as important as bottom-up processing. Because of the strict feedforward architecture of pipelines, the results from components later in the processing sequence cannot retroactively modify the results from earlier stages. Since top-down contributions seem to be crucial to human understanding of language, it is therefore unlikely that pipelines will be able to perform at the human-level.

4.3 Unified Approaches

The most complex type of NLP system architecture is a unified model, in which multiple NLP tasks are combined to create a joint distribution over the final inference. For instance, a part of speech tagger and a parser could be combined so that the overall parse simultaneously maximizes the probability of both components. Thus, unified approaches are capable of simulating some top-down effects, even though they might not have an explicit component that performs that function. Some natural languages, such as Turkish and other agglutinative languages or languages that have a complex morphological structure, such as Arabic, require joint models because the traditional decomposition, say into PoS tagging and parsing, is not a good approximation of their structures (Sarikaya, Afify, Deng, Hakan, & Gao, 2008).

One of the major drawbacks of unified models is that they are computationally expensive, due to the increase in size of the model (Ng & Low, 2004)[9]. This has limited the creation of such systems until recently, as the necessary resources have only now available. Furthermore, unified models in general require more training data, which can be difficult to obtain if it needs to be in the form of a human-labeled corpus. Even though there have been some efforts towards the creation of unified models for full sentence understanding, the resource demands of such models greatly limit their sophistication. A prime example of this is the Markov Logic approach, which trades systematic inference for a randomized, local algorithm (Poon, 2011). Although Markov

[9] C.f. discussion on "all-at-once" vs. "one-at-a-time" experimental results. "All-at-once" refers to a joint model of segmentation and PoS tagging of Chinese text. This method requires 20 minutes to execute, whereas combining separate models for segmentation and PoS tagging require between 1 and 2 minutes.

Logic can be employed to represent such a model in a tractable way, its inherent lack of structured knowledge and inference prevents it from attaining human-levels of performance (Bignoli, Cassimatis, & Murugesan, 2010).

5. Overview of Knowledge-Rich Approaches to NLP

5.1 Introduction

The above sections have: motivated the study of NLP as a useful foray into the frontiers of both Cognitive Science and AI, introduced a number of real-world problem applications for NLP technology, and provided a description of what types of tasks NLP systems are generally designed to solve. Before criticism is cast upon the statistical NLP methodology, it is useful to examine why data-driven approaches became the dominant NLP formalism and why they widely supplanted knowledge-rich NLP platforms. It is also important to briefly examine the shortcomings of knowledge-rich AI (note that this examination has been done in great detail in other publications: (Norvig P. , 2011) (Tugui, 2004) (MacCartney, 2009)[10])

5.2 A Brief History of Knowledge-Rich AI

Until the early 1980's, both NLP research and the AI field in general were dominated by knowledge-rich methodologies. Most early systems involved casting knowledge in the form of prepositions and rules and using some kind of backtracking search algorithm to perform inference. Simon and Newell's General Problem Solver (Newell, Shaw, & Simon, 1959) is probably the most archetypal of this type of system. These approaches relied on the processing of discrete symbols, which could readily be applied to solving formal problems in logic and other branches of mathematics (Newell & Simon, 1956). Eventually, the community of knowledge-based researchers split into two groups—one relying more exclusively on formal logic and inference procedures and another relying more on commonsense structures of world knowledge. The first group, which eventually came to dominate knowledge-rich AI, used methods that culminated in the development of so-called expert systems (Darlington, 1999). The second group focused on commonsense reasoning using representations such as frames (Minsky, 1975) and scripts (Schank & Abelson, 1975), as well as non-monotonic inference techniques, such as circumscription (McCarthy, 1980).

[10] Esp. Section 1.4.2

Commonsense reasoning techniques were explored to attempt to mitigate some of the most difficult epistemic challenges faced while attempting to program an autonomous AI agent (Davis E. , 1990). An example of such an obstacle is the "Frame Problem," which results from having to determine what aspects of the world are altered by a particular event and which ones are not. This can lead to an explosion in the size of reasoning problems if every interaction and non-interaction has to be encoded in detail. Although formal inference procedures may be overwhelmed by this type of situation, it is possible to use laws of commonsense reasoning to circumvent these issues. For instance, the "Commonsense Law of Inertia" stipulates that all aspects of the world remain the same unless if there is a salient reason to believe otherwise (e.g. because of an observed change or from the activation of some specific rule acquired by previous experiences). This axiom has been used to formulate a practical solution to the Frame Problem (Shanahan, 1997).

Although the knowledge-rich approach was initially hailed as a promising route towards human-level AI, it fast became apparent that such techniques were vastly underpowered for the task they were created to perform. The major difficulties of knowledge-rich applications revolve around the fact that such platforms are generally inflexible, exhibit brittleness, and do not scale well from toy problems to real world scenarios. Moreover, the task of combining different knowledge-rich systems into a single unified architecture is not entirely straightforward. Many knowledge-rich frameworks may be inappropriate platforms for supporting a multi-system integration. For instance, reasoning algorithms that are designed to perform efficient inference given some limitations (e.g. Prolog, which is restricted to Horn Clauses (Pereira & Shieber, 2002)) may be useless to the task of integration if one or more of the system components fails to obey those limitations (e.g. a knowledge-based system that allows more general constraints (Liu & Levesque, 2005)).

All of these issues are compounded by the fact that it is difficult to learn highly structured, knowledge-rich relationships automatically from unstructured data. An additional universal difficulty for knowledge-based systems is the problem of belief revision. Although many real-world scenarios require agents to have the ability change their stance on a given item of knowledge, many reasoning systems, such as Prolog, do

not allow beliefs to be revised. There has been a great deal of work towards implementing a knowledge-based system that is also capable of tractable belief revision[11]. For instance, the SNePS project at SUNY Buffalo (SnePS Research Group, 2011), which has been used for NLP applications (Shapiro & Neal, 1982), permits belief revision (Shapiro & Johnson, 2000). However, it has yet to be demonstrated that such systems can be used as the backbone of robust and large-scale systems.

Inflexibility and brittleness in AI systems is caused by the inability to deviate from the knowledge or procedures that are hardcoded a prior into their knowledge bases. This leaves such approaches vulnerable to common situations in real world information processing. For instance, incomplete data, malformed data, and conflicting data all can pose major problems for a program that is unable to adapt itself. In the realm of NLP, incomplete and malformed data appear all of the time in the form of poorly edited, colloquial, or damaged texts. NLP platforms that contain hard constraints about the structure of a grammar will be unable to accommodate sentences that contain either a grammatical error or a construction that was not previously identified by the system's programmers. This non-adaptability was identified by John McCarthy as a major stumbling block for AI fairly early on in the development of the field. Essentially, McCarthy states that in knowledge rich AI, humans do all of the "intelligent" work by identifying the various patterns in data and then coding a fixed solution to recognize and process those patterns.

Without the ability for self-improvement, AI systems will never be able to transcend the level of complexity provided by their human creators. Although it is possible to make some impressive AI systems by direct human intervention, this methodology ultimately results in deficient results for many applications. Consider the example of chess playing. Chess has a large number of fixed rules that can be precisely encoded. These programs simulate expert human chess players by memorizing the best moves after certain board configurations occur and by having a process that can unfold the game tree of to a reasonable level of plies to infer the best move to make regardless of the next move. Through such techniques a great deal of the "non-determinism" created

[11] Note that exact belief revision is tractable in Bayesian reasoning systems if certain restrictions are imposed on the structure of models (Boardman, 2004).

by facing a human opponent can be handled. Computers now rank among the best chess players in the world.

However, it is extremely difficult to encapsulate many significant "real world" reasoning problems from other reasoning domains in the precise way that chess can be. This is largely due to the fact that events in the world are typically not identical (as they would be between different games of chess), but are instead related through some kind of similarity in their structure. One of the reasons why the frames and scripts approaches to AI were largely abandoned was that they were natively incapable of extending themselves to situations that were similar, but not identical, to the situations for which they were designed. For instance, in a script designed to guide an agent through the procedure of ordering at a restaurant, the agent might be encountered with a drink menu first rather than a food menu. If this contingency was not planned for by the script creator, then the whole scenario may be compromised, even if all of the prior and subsequent aspects of the situation had been anticipated.

In Schank and Abelton's seminal paper on scripts (Schank & Abelson, 1975), it was imagined that these types of degenerate situations would be handled by maintaining a list of "what-ifs" that would provide alternatives for every exception to each possible action that would be performed by an agent executing the script. However, neither an automatic method for compiling these "what-ifs" nor a description of how their representation might be structured was given. Because no theory for automatically learning these what-ifs was developed, human programmers would have to not only directly specify the initial script, but would also have to update the system every time it encountered an irreconcilable condition. Furthermore, by Schank and Abelton's own admission, a general-purpose agent would have to be provisioned with a large number of quite specific scripts. Without the aforementioned ability to learn, all of these scripts would have to be hand coded for each new situation that a scripts-based agent would encounter.

This reliance on prefabricated reasoning tools rather than on some mixture of past experience and improvisation is one of the greatest weaknesses of AI—it is also one of the greatest deviations of AI from "naturally intelligent" systems. Selecting the most appropriate script for a particular situation is a difficult problem if there are gaps in the

script coverage. Consider if an agent was equipped with a script for "buying a house" and a script for "ordering from a menu". If the agent was presented with the situation of "selecting and purchasing a car," it might be supposed that the scripts for "buying a house" and "ordering from a menu" could be combined in some way to guide the agent's actions. However, no method for selecting and combining scripts in this way was given by Shank and Abelton. Unlike with the task of recognizing the current (and very definite) state of a chess board, it is unclear how the scripts most relevant to a particular situation would be retrieved from memory. To address this particular issue, a more sophisticated approach, known as "cased-based reasoning (CBR) (Kolodner, 1992)," was created to enable script- and frame-like systems to have greater generalizability. The ability of CBR systems generalize is based on their use of similarity metrics to match the current situation with the stored "situation cases" it is most like. One of the most vehement criticisms of CBR-based systems is that it is difficult to use cases in formal knowledge-rich inference procedures. However, there has been some work on integrating the frames/scripts/CBR AI methods with logical knowledge (Bergmann & Mougouie, 2006).

With regards to NLP, adaptable scripts and frames would be extremely useful for establishing methods for decoding the pragmatic aspects of language (Dahlgren, 1988). By casting inference as a search process, it has been possible to ameliorate some of the shortcomings of purely knowledge engineered systems. One of the benefits of using a search-based methodology is that it enables reasoning systems to construct a solution to a problem that is not necessarily hardcoded into their knowledge base. Instead, searching permits the use of on-line, general reasoning schemes to explore a space of possible solutions, which presents an opportunity for learning from both successes and failures. This allows an agent to search for how to continue in a particular situation if there is no explicitly defined contingency. However, to be most useful, search is often paired with some kind of weighting scheme, which allows the outcomes of various decisions to be analyzed and compared in terms of their expected utility (Cassimatis, Murugesan, & Bignoli, 2009).

Nevertheless, the searching paradigm still requires a system to have some procedure for continuing operations under unforeseen circumstances. While it is almost

certainly infeasible to encode presaged alternatives for every possible situation that an agent might encounter, a more productive approach has been to propose a reasoning substrate in which problems in a wide variety of domains can be decomposed into problems in just a few domains. An example of this is Polyscheme's (Cassimatis et al., 2010) physical substrate hypothesis for explaining the architecture of the human cognitive system. Rather than providing detailed scripts and micro-theories for every domain that might be encountered by a human, such as reasoning about language, reasoning about mathematics, or reasoning about economics, Polyscheme provides detailed mechanisms only for reasoning in physical and social domains.

The motivation behind this approach stems from an evolutionary account of human cognition. Almost the entire evolutionary history of human-level natural intelligences occurred in an environment that required competence in solving physical and social reasoning problems. Therefore, it is sensible that mechanisms that are devoted to such tasks would constitute the greatest share of the human cognitive system. On the other hand, tasks such as reasoning about financial investments would have had a very short existence compared with the total history of cognition and are probably not part of the human genetic inheritance. And yet, humans can nevertheless consistently make economic decisions. Some mechanism must enable such abilities; thus, if reasoning problems in other domains could be mapped onto physical and social reasoning problems, then such a transfer might be useful for designing AI systems. It turns out that this is the case. Much work has been done to establish the feasibility of this mapping, both in the design of cognitive architectures (e.g. Polyscheme (Cassimatis, Murugesan, & Bugajska, 2008)) and in the study of linguistics (e.g. Lakoff and Johnson's work about the physical groundings of metaphor (Lakoff & Johnson, 1980) and work on the Force Dynamics interpretation of Cognitive Grammar (Talmy, 1988)). However, these theories will have to be implemented in an efficient system in order to serve as the basis of AI and NLP technology.

In an attempt to stop-gap the space between the limited hand-crafted expert systems of the 1980s and systems that would be capable of human-level self-directed knowledge acquisition, attention in the field of AI returned to earlier work in commonsense reasoning. Further development of commonsense reasoning occurred in

the form of ambitious knowledge-engineering projects, most famously Lenat's Cyc project (Guha & Lenat, 1990). Cyc invested hundreds of person-hours and millions of dollars in funding towards creating an ontology-based repository of commonsense information. Despite having a large amount of resources at its disposal, the Cyc project has met with only middling success. Gaps in the Cyc ontology must be filled manually by human programmers, since the system is incapable of induction. Humans acquire so a vast amount of knowledge from their environment that it is probably infeasible to explicitly encode all of it upfront. Moreover, there is no available metric to gauge how much of the universe of commonsense knowledge that Cyc has covered, which makes it unlikely that Cyc could be used as a basis for un-encoded facts in every domain. An additional problem is that Cyc has no probabilistic interpretation; its statements are either wholly true or false. Since commonsense reasoning may be contradictory and subject to possible exceptions under certain circumstances, the basic design principles of Cyc are not conducive to its overarching goals. Generating exceptions is one of the most difficult reasoning tasks. The volume of possible exceptions to even seemingly straightforward facts[12] is immense. Therefore, it is probably necessary to derive exceptions from "default facts" and some domain-general knowledge that can apply to many aspects of reality[13].

Exceptions are closely related to an epistemological issue known as "The Qualification Problem (Elkan, 1995)," which has made it difficult for researchers to create AI programs that exhibit robustness under real-world circumstances. Indeed, explosions in the number cases required for real-world applications was a problem repeatedly encountered during the golden age of AI, when systems were developed to solve small-scale "toy problems" with more impressive real-world counterparts. A prime example of this phenomenon is Terry Winograd's SHRDLU program (Winograd, 1971) that employed natural language understanding to perform a toy problem called "blocks world." Blocks world involves the manipulation of a small number of colored blocks by

---

[12] E.g. A bird can fly unless: it is a nestling, is injured, is deformed, is sick, is tranquilized, is caged, is dead, is currently underwater, in high winds, is a flightless bird, is a model of a bird, is a picture of a bird, is something that is being called a bird metaphorically, etc.

[13] E.g. In the "bird" example given in foot note 7, the default fact might be that "All birds can fly (unless if a salient relevant exception holds)." A salient relevant exception might be: "dead, sick, deformed, and injured **animals** often cannot perform the default behaviors associated with them." This would be retrievable via a spreading of activation scheme if it is known that the current focus of reasoning is a bird, that a bird is a type of animal, and that this particular bird is injured, deformed, sick, dead, etc.

a robotic actuator. Although some of the commands that SHRDLU was able to process seem impressive, they are predicated on a large number of assumptions: the uniqueness of objects, the limitation on the number of actions and objects, and the expectation that all actions will result exactly in a predefined change of state.

Even the expectation that a particular action will be executed precisely is a bad assumption, as was discovered in early robotics systems. These systems assumed that each command to a servo would result in an exact motion. However, interference from friction and other mechanical forces causes a wide variability in the results of executing those commands. The "Shakey" robotics system is a good example both of how uncertainty can result in great difficulties for classical AI methods (Flynn, 1985)[14] and how such deficiencies can be covered up by effective manipulation of public presentation (SRI, 2010).

In terms of using learning to extend knowledge-rich systems, it has been difficult to make satisfactory progress. Because of this difficulty, these systems are very likely to be handcrafted by dedicated human knowledge engineers. Of course, some NLP subproblems require a vast amount of hand-coded knowledge to be solved. An example of this is PoS tagging. The leading knowledge-rich system for PoS tagging is the English Constraint Grammar (Karlsson, Voutilainen, Heikkilä, & Anttila, 1995), which contains approximately 10,000 human generated rules to express the patterns of PoS usage in English text. Even though this system is almost exhaustive (it is created by people examining a large number of examples in an attempt to write down all of the general rules and exceptions to PoS use), it is not able to obtain perfect accuracy. One of the reasons for this discrepancy is that, as mentioned above, some parts of speech are ambiguous without taking into account a context wider than the current sentence that is being analyzed. The accuracy of the English Constraint Grammar PoS tagging system has been reported to be as high as 99% (Tapanainen & Voutilainen, 1994), which is achieved by taking into account ambiguous PoS taggings, as well as a statistical "fall back" contingency in case if no constraints can be definitively applied for a given input.

Despite this extremely high performance, Constraint Grammar-based PoS tagging is rarely used, since systems that are much simpler to compose (i.e. those that learn part

[14] C.f. Section 2.1 – "Shakey 1967-1969"

of speech patterns automatically) can achieve almost equivalent results. Moreover, the platforms that employ a Constraint Grammar-type "expert system" are difficult to maintain and extend. If a new PoS pattern is encountered, an expert system-like Constraint Grammar would require a significant amount of human made modification. One solution to this problem would be to create a theory of inductive learning that would be sufficiently powerful to learn the types of specific constraints upon which Constraint Grammar relies. It is possible to create programs that can induce complex, fuzzy concepts from an appropriate selection of exemplars--Patrick Winston developed a system that was capable of learning the abstract structure of a geometric arch (Winston, 1970). However, in practice, it is not only difficult to learn complex relations, but it is also difficult to even gather a large enough bank of learning examples for such a task. Obtaining large hand-annotated corpora, especially with knowledge-rich structures, is very time consuming and expensive (Ide & Brew, 2000). There have been attempts to populate the content of knowledge-based systems by using data sources with varying levels of structure (Tribble & Rose, 2006). However, the ultimate goal of these systems is to be able to ingest information using human-level natural language, which is a requirement far from being fulfilled by the current state-of-the-art.

A number of NLP systems have been implemented using deductive, structured-knowledge approaches to problem solving. One of the most common formalisms used to study language is a Finite State Automaton (FSA), which can solve the recognition problem for languages that belong to the Regular Languages level on the Chomsky hierarchy. Although Chomsky has shown that natural languages are generally not well-described by Regular Language patterns, there have been several applications for FSAs in NLP research. First, FSAs can be used to solve problems that do not involve full syntactical parsing. Thus, they have been applied to such tasks as morphological analysis (also known as stemming) (Karttunen & Beesley, 2005). Deterministic approaches to PoS tagging have used Finite State Transducers as the underlying computational mechanism (Roche & Schabes, 1995). FSA-based PoS tagging systems can attain a tagging precision of 95%. Alternative methods for PoS tagging, namely "decision tree learning (Orphanos, Kalles, & Papagelis, 1999)" and "transformation based learning (Brill, 1995)" techniques produce comparable results.

Beyond PoS tagging, the linking of several layers of FSAs into a cascade (Abney, 1996), in which the output of one level of FSAs is passed as input to a higher level, enables recursive phrase structures to be approximately parsed. Recall that recursive phrases are the primary features that cause natural languages to extend beyond the domain of Regular Languages. The "cascade" can be unpacked as a sequence of rules that map an input sentence to a phrase structure. This type of system is intended to perform shallow parsing or "phrase chunking," which is arguably a simpler problem than full syntactic parsing. On the "chunking task," finite cascade systems are able to approximate human performance with their per-word normalized accuracy of 92.1%.

Structured knowledge approaches can also be leveraged to do full parsing. For instance, LR parsers (Knuth, 1965) are able to apply constraints in a systematic way to decode the latent structure of a string with a set of context-free or nearly context-free (i.e. Tree Adjoining Grammars (Joshi & Schabes, 1997)) grammatical rules. Applied to sentences in the Penn Treebank corpus, LR parsers obtain approximately 77.78% accuracy (Prolo, 2002). Even though these systems have been shown to be implementable, knowledge-rich and highly structured systems are difficult to create and maintain. This investment comes with little payoff, as knowledge-lean approaches, as will be shown in the sections below, are capable of matching or outperforming their counterparts.

6. Rise of Machine Learning

At the same time that structured knowledge-based, expert systems-like AI methods were being developed, systems based on other approaches to modeling reason were explored. These approaches belong to the machine-learning or "knowledge-lean" tradition. In actuality, these methods are somewhat based on knowledge, too, but with two main differences that separate them from the classical AI systems described above.

First, knowledge-lean approaches essentially lack structured, explicit, or relational encodings. Through the use of machine learning algorithms, knowledge-lean systems perform parameter fitting to model a function that maps from an input signal to an output signal. For instance, one such function might be one that maps from a sequence of English words (e.g. a sentence) to a sequence of English PoS tags. The learning performed by these systems can be accomplished in a supervised, semi-supervised, or unsupervised environment. Supervised learning involves using a "teacher" signal that can judge whether the output produced by a program is correct. This signal, which is usually presented in the form of labeled training data, is used in some way to adjust the internal parameters of the program (e.g. the back-propagation algorithm for neural networks. Semi-supervised[15] learning involves a mixture of labeled and unlabeled cases, whereas unsupervised learning does not rely on annotation. The benefit of using supervised data is that more complex functions can be learned; however, this comes at the cost of requiring annotated training data. As was noted above, the cost of producing marked corpora is high enough that unsupervised methods have gained in popularity (Vlachos, 2011).

Second, since knowledge-lean systems rely mainly on the ability to induce their knowledge content from some kind of information signal, their knowledge tends to have a very simple structure, when compared to what knowledge-rich approaches are capable of supporting. As the complexity of the function to be learned increases, the amount of data required for training also increases; the computational requirements of learning, in terms of space and time, may also increase. However, by examining the results produced by knowledge-lean vs. knowledge-rich programs, it seems that a low-level of

[15] Semi-supervised learning sometimes involves algorithms that rely on "self-teaching" techniques in which a program is responsible for generating its own labels for training data (Scudder, 1965).

representational complexity coupled with a potent learning algorithm can perform just as well as a system that was created by a dedicated human knowledge engineer. For instance, consider the problem of PoS tagging. Knowledge-lean systems are capable of achieving individual word tagging accuracies between 97% and 98%. Although the Constraint Grammar system mentioned above claims up to 99% accuracy, in practice it achieves closer to 96%, once ambiguous PoS taggings are discounted. Due to the high volume of data that is processed during a PoS tagger evaluation, even small differences in performance accuracy are statistically significant—this fact has led to an accuracy "arms-race" of sorts within the PoS tagging community. Ultimately, factors beyond accuracy measures have been more instrumental in establishing the current trend of favoring knowledge-lean approaches in the AI and NLP field. For instance, the deployment of knowledge-lean systems is much more straightforward than knowledge-rich systems in many cases.

Although applying knowledge-lean frameworks to novel situations may require some modification (e.g. some techniques that work well in English have to be modified to perform adequately with other languages), this typically requires far less technical expertise that it would be needed to develop a new tailored knowledge-engineered system. As was mentioned above regarding Constraint Grammar, developing a hand coded PoS tagger for English requires an exhaustive analysis of English texts to mine all of the patterns of usage for individual PoS tags. On the other hand, knowledge-lean, machine learning algorithms are capable of extracting these patterns automatically. Thus, the same technology, once developed, can be repurposed in many ways, which greatly reduces the turn-around time of creating new AI or NLP programs. This increase in efficiency comes at the cost of making some assumptions that make the learning process tractable. The major upshot of these limitations is that the patterns that are induced by knowledge-lean approaches are largely fixed (i.e. new types of patterns are difficult or impossible to learn) and fairly basic (e.g. a knowledge-lean platform might be limited to patterns involved word co-occurrence). Nevertheless, at least in English, simple patterns can account for a large portion of the signal in natural language. In light of this, it is not surprising that knowledge-lean approaches have become popular—they appear to be far more labor- and cost- efficient for their performance return. Additionally, even though

knowledge-lean approaches are inflexible in terms of not being able to incorporate new types of patterns during learning, they are capable of learning new instances of the pattern types they were programmed to recognize. This property of adaptability allows knowledge-lean approaches to be seen, at least at the offset, as more robust than knowledge-rich approaches.

Despite the apparent advantages of the knowledge-lean paradigm, they did not gain widespread acceptance until the late 1980's to the early 1990's. Although machine-learning style methods were developed in tandem with their knowledge-engineered counterparts from the inception of the field of AI, limitations in computing hardware were responsible for forcing the fields of AI and NLP to focus mostly on expert-system like approaches. However, once computing technology advanced to the point where affordable systems could support large-scale machine learning algorithms, the knowledge-lean methodology quickly achieved its current status as the state-of-the-art. Knowledge-rich approaches, which may rely on computationally complex inference routines, were not able to scale up as well as their knowledge-lean counterparts. Presently, there are roughly two main families of knowledge-lean systems, each of which derived were from one of two drives towards harnessing machine learning to automatically produce computer programs of high complexity.

The first of these lineages dates from before the advent of modern computers. By the early 20th century, statistical models were developed to characterize the distributions of phenomena that were of interest to scientists and engineers. One of the most basic of these modeling techniques is known as linear regression (Edwards, 1976), which attempts to fit a linear equation to a collection of data in the hopes that the resulting line will reveal some insight into the hidden function that generated the data points. A related technique is logistic regression (Hilbe, 2009). As will be seen in the section on Maximum Entropy modeling, Maximum Entropy models, a popular knowledge-lean paradigm, are based on a logistic regression paradigm. In economics, mathematical models were created to simulate complex cost optimization problems through techniques such as linear programming (Murty, 1983). Linear programming problems can be solved with numerical analysis techniques; likewise, Maximum Entropy models are trained by related techniques in order to optimize their parameters (Gamboa & Gzyl, 1990).

Other modeling techniques that attempt to derive properties about underlying generating functions include principal component analysis (Jolliffe, 2005). Principal component analysis works by projecting data recorded on some original dimensions onto a set of new dimensions such that each successive projected dimension correlates to a hypothetical variable that can explain the greatest amount of variance in the data after taking the previous components into account. The strategy of mapping data onto new dimensions is a critical part of a knowledge-lean approach known as a Support Vector Machine (SVM) (Steinwart & Christmann, 2008). This remapping enables SVMs to learn non-linear functions. Early efforts to model stochastic processes (Doob, 1953) that involved the passage of time (as opposed to a set of static data points) were direct precursors to the family of Markov Model formalisms. Markov Models are very common machine-learning platforms that have been developed extensively since the mid-20th century. An entire future section will be dedicated to an in-depth discussion of Markov Models with respect to their role in NLP modeling.

The second tradition of knowledge-lean models was inspired by the idea that an artificial computing device might benefit from mimicking the style of biological computations. Neurological experiments revealed that the brain appeared to perform its functions through controlling the flow of electrical impulses that transmit along networks of simple cellular processing units called neurons. Artificial Neural Networks (ANNs) (Rojas, 1996)were created to emulate this style of computation. Because computation occurs as a function of the connections between simple units rather than through the instruction of a powerful central unit, these systems became known as Connectionist models. In the case of ANNs, computation is achieved by submitting a signal to the input layer neurons of the network, which then propagates forward via weighted connections that link to subsequent layers and so on until the output layer is reached. Each connection attenuates the signal passing through it according to the weight on the connection; each neuron sums all of the incoming signals and then either fires or does not fire a signal depending on whether the input sum exceeds that neuron's threshold. It is most common for ANNs to earn via a "backpropagation" algorithm (Rumelhart, Hinton, & Williams, 2002) that compares the actual output of the network with the expected result. By examining the difference in value for each neuron in the output layer, it is

possible to adjust the weights leading into those neurons; this process is repeated sequentially for all of the layers in the network by examining how the previous weights were modified. The requirement of a learning signal therefore makes backpropagation an inherently supervised algorithm.

There are many variations of Connectionist models that can be distinguished by their computational properties. One of the first and simplest Connectionist paradigms is a Perceptron network (Rosenblatt, 1958), which attempts to learn an optimal way to partition a dataset via a hyperplane. The so-called Connectionist approaches, however, fell out of favor early on after the book "Perceptrons" demonstrated in 1969 that single-layer perceptrons could not learn linearly inseparable patterns, such as the XOR function, from data (Minsky & Papert, 1969). Later elaborations on Connectionist approaches enabled them to learn non-linear data, which lead to their eventual resurgence. Some of the more notable types of systems are the multi-layer perceptron (Hornik, 1991), which has the capacity to learn non-linear partitions of data, and recurrent neural networks (Pineda, 1987), which incorporate a signal-looping layer that can serve as a kind of memory, thus allowing the modeling of dynamic generating functions.

Work on perceptrons ultimately inspired what are known as Kernel Methods (Shawe-Taylor & Cristianini, 2004) for machine learning. Kernel Methods define a transformation that maps data points onto a new dimensional basis, which allows the data to be more easily partitioned. Support Vector Machines (SVMs) (Cortes & Vapnik, 1995)are one of the most widely spread types of Kernel Machines and have been used in a number of NLP problems that rely on the classification of documents, such as the detection of spam emails (Joachims, 1998). It is important to note that following a particular path of development does not lead to mutually exclusive sets of technologies; SVNs can be seen as a derivate of either of the two types of knowledge-lean traditions that were discussed here.

7. Critiques of Mature State-of-the-Art NLP Platforms

In order to anchor the criticisms that can be levied against individual statistical, machine-learning-type NLP methods, as well as the criticisms that have been made against knowledge-rich approaches, it is instructive to examine the actual performance of products and services that are derived largely from such methods. There are a wide range of commercial systems that have recently been developed with the aim of making progress towards human-level NLP. Some of these systems maintain the knowledge-lean bias that is dominant in most academic NLP groups; other incorporate knowledge-rich components to varying degrees. Each of these systems decomposes the processing of language in different ways; some take a more holistic view of the parsing process than others. In extreme cases, even though the dissection of language processing into discrete stages may be cognitively implausible and the reported evaluations of statistical NLP approaches may be misleading, in the end, what matters the most is whether or not these technologies, given sufficiently broad intellectual and financial support, can actually serve as the foundation of systems that can achieve human-level performance. It is not even necessary to produce a fully human-level system immediately or even one that is geared towards solving the NLP problem in general, only that the potential for such an outcome is present.

As with all technologies, it is necessary to accept that there are certain tasks that a given tool is useful for performing and many other tasks that it is not. Thus, the bar is set rather low for the NLP field. There only need be some large scale project (certainly the commercial gains alone are large enough to ensure that investments in this area will continue to be made) that has the potential to evolve to increasingly complex linguistic capacity. While it is possible and potentially desirable to engineer intermediate technologies systems that are profitable, there are so many applications for full-blown NLP technology that it seems improbable that one or more major corporations would not have developed such systems if they were truly in reach of the present state-of-the-art formalisms. Perhaps, even more than the revelation of the misleading performance reporting methods that hinder the field, the non-existence of even marginally advanced large-scale NLP systems is telling of the shortcomings of the current practices. By examining the most prolific of real-world large-scale NLP projects, it becomes clear that

these systems are not very promising steppingstones to human-level language understanding.

7.1 Google Search

There is an ongoing blog published by Google (Google, 2013) that documents its research and development efforts, which heavily involve extending the sophistication of the NLP technology used by its search engine front-end. One entry on Google's blog, called "Helping computers understand language" (Google, 2010), illustrates how context-sensitive synonyms can be used to augment the matches that are returned by a search engine. Some of the most compelling examples on this page demonstrate how the Google interface can interpret the acronym *GM* in a large number of ways based upon context (e.g. *GM + car = General Motors*, *GM + food = Genetically Modified*, *GM + navy = Gunner's Mate*, etc.). Testing these queries on Google.com reveals that they work as advertised, at least in the sense that the top search results are relevant to the usage of *GM* in the context given by the rest of the search query. However, it is not hard to muddle the results by asking for a query with terms that are associated with different contexts. Searching for *GM recall allergy* returns a story about genetically modified corn as the top result, but all of the other hits on the first page of results pertain to GM vehicle recalls. On the pages that deal with GM vehicle recalls, all of the forms of the word "allergy" appear mainly in links to other websites that have nothing to do with the search query. Alternatively, the sites on the first page of results for *ethanol-fuel GM* contain text in which GM refers to General Motors, not Genetically Modified, even though one would expect some hits for Genetically Modified corn to be retrieved. This type of performance suggests that whatever processing is being done by Google is largely supplemented by the traditional search engine practice of finding highly ranked documents that contain the words in the search query. Certainly any actual NLP is done on the query string and that appears to be very limited.

In order to determine how useful this type of natural-language-based search modification really is, it would be ideal to compare the results from a "standard" search engine that is not augmented with NLP technology. Ironically, searching for *search engine without natural language* is not helpful in finding such a service; rather the results

pertain to search engines that do use natural language processing techniques. One major difficulty that is faced with the prospect of using deep NLP techniques to refine search results is the fact that the raw search query is almost always delivered in a form that is poorly structured for direct syntactic or semantic processing (Sullivan, 2006). Many search strings consist of few words, often only one, and therefore lack context for what the user is attempting to find. This lack of structure requires search services to fall back onto statistical measures of query popularity that have little or nothing to do with natural language use. Therefore, even though there are instances in which the interests of the user aligns with the most common searches that are similar to the original query, there are enough difficult cases to continue to frustrate the searching process.

However, finding relevant pages is only the first part of the information retrieval process. Mining texts for their content is usually left to humans to perform, but Google has publicized that it is capable of reporting quick answers for certain queries. These answers are based on extracting what is determining to be the most relevant information related to the search query from the documents that are retrieved by the search engine. Simple questions, indeed, can be answered. For instance, the query *when was Lincoln born* will return *February 12, 1809*, having made the assumption that the *Lincoln* in the query is President Abraham Lincoln. Likewise, *who was Lincoln's wife* will return the expected answer of *Mary Todd Lincoln*. A more complicated query, *when was Lincoln's wife born* returns no such quick answer—the user is left to extract that information from the pages that are pulled up in the normal search results.

Because Google's NLP techniques are focused on statistical, knowledge-lean approaches, they have been able to exploit the low cost and high adaptability of that technology to rapidly produce improvements to their search service. However, the resulting benefit to users is still fairly low; Google is not capable of understanding even moderately complex grammatical structures in its queries, as is indicated above. If queries cannot be deeply processed by Google's methods, it is also unlikely that Google would be able to parse the raw documents that it indexes, especially if that service had to be performed in real time. It can be argued that the purpose of a search engine is solely to provide suggestions about which documents might be relevant and not to be a

question-answering service. Yet, without employing more powerful NLP techniques, the quality of those suggestions will remain low.

Knowledge-lean approaches are inherently ill-suited for representing the complex relational structures that appear in search queries. Despite this, Google's focus on the statistical approach to NLP is unlikely to change for several reasons: 1.) Peter Norvig, who is the Director of Research for Google,[16] is a strong proponent of machine learning-based AI, 2.) Google has access to a vast bank of unstructured documents that enable it to perform very large-scale data mining operations, 3.) statistical approaches enable the fast development of narrowly focused technologies that can provide small improvements in search results; natural language search is extremely rudimentary at this point and so almost any improvement can appear significant.

7.2 IBM's Watson Question and Answering System

Although IBM has a long history of and large investment in developing in-house NLP research and associated derivative technologies (c.f. (IBM, 2004)), one of its most recent projects, known as Watson, has been the most promoted as a precursor to an ultimate solution to the problem of enabling machines to understand natural language. Watson is one of the most widely recognizable and publicized NLP projects ever created, in least in terms of its popular press coverage. This status was achieved when IBM contrived to run Watson as a contestant in a special series of Jeopardy matches against prolific human players from recent years. Like past technology-centered PR campaigns launched by IBM, of which the Deep Blue chess-playing supercomputer that was successfully pitted against then-World Champion Gary Kasparov in 1997 is the best known (Newborn & Newborn, 1997), IBM sought to demonstrate the intelligence of Watson by having it compete directly against humans.

In terms of modern publicity, NLP applications have a distinct advantage over a game like chess, which is mostly followed by a small group of enthusiasts. It used to be the case that highly-abstract contests such as chess were considered to be the hallmark of human thought, but this notion has decreased in popularity, since computers can be competent chess players, but are largely incapable of what most laypeople would

[16] At the time that this document was created (circa Spring 2013).

considered to be basic human activities. Because natural language communication is so pervasive and universal in human behavior, it was once overlooked as a serious subject for inquiries into the nature of intelligence. Yet, years of dedicated NLP research have not produced a system that can even approximate the performance of an average human toddler[17]. As the frequency of human-computer interactions increases, the need for better interfaces has become apparent. Many automated services would be much more efficient if they could be controlled by natural language. For these reasons, NLP has become the new standard for gauging computer intelligence. However, one of the potential troubles for the NLP field is its tendency to equate truly intelligent behavior with proficiency in artificial, narrowly defined and limited tasks.

Because of the susceptibility of the NLP field to hyperbole rather than meaningful results, it is necessary to access how well the Watson system is truly capable of performing NLP tasks in general. Indeed, Watson is capable of handling a massive throughput of information, which enables it to analyze vast amounts of text-based sources. Moreover, the system is capable of learning from both its successes and failures as it seeks to refine its knowledge base. These attributes appear to make Watson a prime candidate for human-level NLP: it is capable of storing large amounts of diverse information and it is capable of self-monitoring and modification. Much was made about Watson's crushing victory on a special series of *Jeopardy!* Episodes (Jeopardy, 2011).

However, there is no reason to equate success on *Jeopardy!* with success in an a realistic, real-world environment for NLP. It turns out that the categories of knowledge and the forms of questions on *Jeopardy!* are highly constrained to a relatively small number of forms, in comparison with the diversity of human-level natural language use. In fact, *Jeopardy!* question patterns are so stylized that one of the greatest human *Jeopardy!* players of all time used data mining techniques to selectively train himself on the most valuable *Jeopardy!* question categories (see: (NPR Staff, 2011)and (Jones, 2012)). If a human could make use of these statistics, which have little to do with natural languages, it follows that a powerful computing system could likewise do so. Without having had the need to adapt beyond the *Jeopardy!* environment, there is no reason to

[17] Not counting contrived tasks, such as part of speech tagging done in isolation of other linguistic processing.

assume that Watson's architecture will truly be useful for more general types of NLP situations.

Accordingly, popular reaction to Watson's dominance of its human opponents in Jeopardy was not universally met with praise. Indeed, philosopher John Searle disputes the notion that Watson had any true cognitive sophistication (Searle, 2011) in a reissue of his infamous "Chinese Room" thought experiment (Searle, 1980), which claims that computer-like symbolic processing systems are incapable of deep understanding. Regardless of whether Searle's assessment is valid, a close examination of Watson's Jeopardy performance reveals that, despite an excellent overall score, the system could not have had a particularly deep understanding of the semantic content of the questions. This observation is strongly supported by Watson's more prominent blunder: answering "What is Toronto?" for the clue "Its largest airport was named for a World War II hero; its second largest, for a World War II battle." (IBM, n.d.). In an interview conducted after Watson's victory (Silverman, 2011), Doug Lenat, a computer scientist known for his leading role in the CYC knowledge engineering project, illustrates that the fact that Watson relies heavily on statistical analysis of language and statistical reasoning prevents it from doing well on structured reasoning tasks that require "common sense knowledge." For instance, it is clear that Watson had no true conceptual model of geopolitics or even the basic properties of physical containment from its "Toronto" answer.

7.3 Apple's Siri Personal Assistant Software

Siri is one of the most highly publicized NLP applications: the fact that many of the commercials for Siri feature prominent celebrities underscores the public allure of having access to a machine that can react as fluidly to a situation as a person (Rose, 2012). A large amount of funding and effort has been spent on developing and promoting Siri, which should have made its success a high priority for the companies that were involved in its creation. As one of the showcase applications for the Apple iPhone, Siri is a high stakes project—if any NLP system could be expected to approach human-level performance, it would be Siri. The cost of failure in this area would be devastating to the reputations of these companies; indeed, negative reviews of Siri have been quite critical of Apple (Honan, 2001). In addition to commercial interests, Siri has a long

history of development in which the organizations involved with its creation were invested in scientifically significant NLP technology. Spun off from a government defense research project called PAL (Personal Assistant that Learns) (SRI, 2010), Siri purports to apply state-of-the art NLP comprehension to boost performance over other available NLP frontends. In particular, Siri is designed to answer questions from a user, with the intent of providing a more accurate answer than a traditional search query.

However, like all of the other platforms discussed in this section, Siri is extremely limited in what it can do. The abilities that Siri has (e.g. checking the weather) are hand coded by application developers—an API is released to allow private interests to extend Siri's reach to their products and services. This leads to a brittleness in Siri's handling of queries that is reminiscent of the expert systems generation of AI and NLP programs. One of the major downsides of the Siri application is that it works primarily by a vocal command interface; speech is notoriously difficult to process correctly based solely on its physical waveform. Therefore, the fact that Siri lacks any top-down knowledge of syntax or semantics makes it impossible for it to make suggestions to correct a poorly-understood query, which is something a human personal assistant would have no difficulty doing. Furthermore, it seems unlikely that Siri will ever become significantly more robust, as Siri itself is based on a central processor that handles the queries (Nusca, 2011). Although it may be possible for Apple employees to modify this system, Siri is incapable of learning or extending itself automatically. Many queries posed to Siri result in falling back to a Google or Wolfram Alpha search. The NLP shortcomings of both Google and Wolfram Alpha are discussed in detail elsewhere in this section, so this relying on those services is known not to be a satisfactory replacement for actual NLP technology. Although Siri is capable of performing some of the duties of a personal assistant, in general, many users see the system merely as a gimmick (Manjoo, 2012). Ultimately, humans must be given the final decision on whether any NLP technology is truly useful for making tasks more efficient.

As of 14 NOV 2012, experiments to test Siri's language comprehension skills[18] revealed a number of commands for which it was unable to arrive at a correct response. These errors mostly involve sentences or phrases that have a complex syntactic and

[18] Data generously provided by Professor Nicholas Cassimatis.

semantic structure. An example of these mishandled commands in the application of searching for Twitter posts follows, along with Siri's actual responses, and an explanation of why the responses are incorrect can be found in Table 1.

Table 1. Queries to Siri with responses and analysis of errors in response.

Natural Language Command to Siri	*Generated Search String*	*Error Explanation*
tweets from nasa with pictures	from:nasa	Ignores Tweet content
tweets from nasa about curiosity	from:nasa	Ignores Tweet subject
tweets from nasa last week	from:nasa	Ignores Tweet time
tweets about romney last week	"romney+last+week"	Does not parse time specification as metadata
tweets about romney from last week	"romney+from+last+week"	Does not parse time specification as metadata
yesterday's tweets about the debate	"the debate"	Ignores Tweet time
tweets with pictures about nasa	(tries to send a tweet)	Should have performed a search, not Tweeted
tweets about nasa with pictures	"NASA with pictures"	"With pictures" should in for metadata, not be part of the text search
search twitter for pictures of nasa	"NASA"	Ignores request for pictures
tweets about nasa or debate	nasa+or+debate (no results)	Ignores disjunction

In light of the above performance data, it is important note that formal procedures do exist for compiling meaning from words (c.f. (Kearns K. , 2000)). These methods operate in a similar way to grammatical phrase construction and could, in principle, be programmed into an NLP system, which would allow many of the errors described above

to be avoided. In order for an NLP application to be truly useful, it will need to incorporate a much more sophisticated parsing and interpretation scheme than Siri currently employs.

7.4 Wolfram|Alpha's Computational Knowledge Engine

Another much-hyped and promoted Internet-based service that utilizes NLP technology, Wolfram|Alpha purports to be an alternative to standard search engines, such as Google. Rather than find individual documents that match some query, Wolfram|Alpha aims to answer specific questions that are posed to it. This process requires at least the following steps: parsing the query, locating and aggregating the appropriate data, computing whatever information was requested, and displaying the resulting information to the user. Currently, Wolfram|Alpha employs NLP techniques, which are only vaguely described by the developer's company in the product's FAQ section (WolframAlpha, n.d.), exclusively in the parsing of the user's query (as opposed to being used for the parsing of documents and databases).

Although there is a statement to the effect that innovative, advanced algorithms for NLP are used in Wolfram|Alpha, as some examples below will demonstrate, it has a very limited command of language. Wolfram|Alpha does not engage in any self-directed or automatic acquisition of knowledge, but instead relies upon human knowledge engineers to hand input new facts and computational models for specific domains. Unlike most search engines, no aspect of Wolfram|Alpha's knowledge compilation is obtained from open information available on the Internet and is instead powered by the use of Wolfram|Alpha's proprietary API applied to specific data sets by individual concerns. However, as the developer points out, the engine does learn in a limited sense, as each answer that it provides is retained by the system. There are plans to enable Wolfram|Alpha to adapt its processing models via some automatic learning process, but no details have been provided regarding this effort. Mathimatica (Wolfram, n.d.), an engine for mathematical programming, provides the means for Wolfram|Alpha to compute the answers that it provides. There does not seem to be any sense in which the system achieves any understanding of the tasks being completed.

Despite being touted as a potential competitor for a portion of Google's traffic (Pollette, n.d.), Wolfram|Alpha is not very adept at answering many complex or compound queries. The two greatest shortcomings of the Wolfram|Alpha platform that contribute to this low level of performance are: 1.) its unsophisticated parsing of input queries and 2.) its inability to self-adapt to new requests or situations. Presumably, it would be possible for Wolfram|Alpha to improve the handling of complex queries; nevertheless, the facts that Wolfram|Alpha requires humans to curate its knowledge base and that the system appears to utilize a minimum of semantic knowledge are more severe limitations.

As an illustration of how well Wolfram|Alpha performs, the following queries were directly entered into the system. It is clear that the service is unable to handle queries with more than rudimentary semantic or linguistic content. For instance, the parser is able to understand *When was Lincoln born?* as a request to obtain the birthday of U.S. President Abraham Lincoln. However, when a more complicated construction is used to describe the subject of the query, the system fails. *When was Lincoln's wife born?* falls back to the same underlying query that provides Abraham Lincoln's birthday. While Wolfram|Alpha is capable of taking some context into account during the parsing process, it ignores any part of a query that it cannot recognize. Thus, NYC is correctly recognized as the city of New York, New York in the query: *What is the weather in NYC?* This request displays the current weather, along with the weather for the next few days in the appropriate location. Yet, identical results are returned when the query is changed to: *What will the weather be in NYC next week?*

Additionally, the types of information that Wolfram|Alpha can process is limited by the current extent of its ontology. Despite having knowledge about a vast number of topics and entities, it is unable to answer a question that extends beyond what it is hardcoded to provide. In the realm of mythology, Wolfram|Alpha correctly identifies Troy as the answer to *Where did Achilles die?* This answer is available because the system is designed to recognize Achilles as a person and that a person has a location for their death, among several other common facts. The boundaries of Wolfram|Alpha's knowledge can easily be surpassed in many of the domains in which it claims

competence. If asked *When did Zeus turn into a Bull?*, the engine returns gives results related only to the word "turn."

Interestingly, Siri and Wolfram|Alpha have a partnership that has a large impact on both of the products. Siri uses Wolfram|Alpha to answer many of its queries (Wikpedia, 2013), to the extent that traffic from Siri comprises approximately 25% of Wolfram|Alpha's total load (Aimonetti, 2012). Because of this significant overlap, Siri's linguistic capacity is likely to be significantly limited by the capabilities of Wolfram|Alpha. Moreover, Wolfram|Alpha may decide to develop itself as a niche tool for Siri, rather than to pursue more general operation as an AI agent.

7.5 Microsoft's Powerset/Bing and Cognition Search Engine Augmentation

The Cognition company (Cognition, n.d.) advertises that it possesses the world's most semantically-advanced NLP system, which is based on what it calls a semantic map of the English language. This semantic map contains relational information about the various contexts that are associated with a large number of English words and some of the most commonly used English phrases. Cognition states that this database is the most complete repository of its kind. Additionally, Cognition claims to take into account the syntactic properties of sentences (e.g. distinguishing between nouns and verbs in a query) and morphological permutations (e.g. different inflections of words, as well as alternative spellings and common misspellings). These features are encouraging, as they are certainly necessary building blocks for realizing a human-level NLP platform, but, again the measurement of a tool's usefulness lies in its ability to perform—the Cognition system's overall performance is no better than Google's.

Since Microsoft's Bing (Bing, n.d.) search service is powered in part by Cognition's technology, it is possible to demonstrate this fact by running a simple search request on Bing. Testing out the example *When was Lincoln's wife born?* on Bing reveals the service's inability to process even mildly complex syntax or semantic relations. Powerset (Wikipedia, 2013) was an independent search augmentation tool that was somewhat similar to Cognition, at least in its aim to enable NLP-enabled search capabilities (e.g. for searching Wikipedia). Now Powerset has been acquired by

Microsoft (Marshall, 2008) and has been folded into the Bing search engine. Moreover, Cognition has itself generated a document criticizing Powerset (Cognition, 2008), which also describes how Cognition is capable of outperforming Powerset in many situations. Due to the law of transitivity, if Powerset is less powerful than Cognition and Cognition is nowhere near the human-level of performance, then it follows that Powerset is also short of the mark.

7.6 N.E.L.L. (Never-Ending Language Learning) Project

Researchers at Carnegie Mellon University launched a project, known as N.E.L.L. (Carlson et al., 2010) to develop a system that was capable of autonomously generating a database of commonsense world knowledge using unprocessed text available on the Internet. This is a step in the direction of producing self-adapting and self-directed programs; the development of such systems is considered by many AI researchers to be essential to the creation of human-level artificial reasoners. Moreover, the relatively high expense of creating commonsense knowledge and the extreme importance of commonsense knowledge, both to linguistic and to non-linguistic reasoning tasks, makes it essential to produce large, tractably accessible knowledge bases for commonsense reasoning. Projects, such as CYC (Lenat & Guha, 2993), have attempted to hand-code commonsense knowledge. Yet, this approach has so far been unable to produce coverage for general knowledge that is sufficiently complete to solve non-trivial problems. Few applications of CYC have been released, despite the millions of dollars that have been put into its funding. Thus, it is clear that projects like N.E.L.L. would be valuable for progressing towards the goal of human-level language understanding.

Several aspects of N.E.L.L. suggest that this system has the potential to learn many of the nuances of natural language without the need for large amounts of human intervention. N.E.L.L. is based on a semi-supervised learning algorithm, which enables it to extrapolate some kinds of new relationships that are not explicitly encoded in its initial configuration. Yet, there are several major drawbacks to the N.E.L.L. system. Primarily, although it employs some sophistication in the structure of its inductive process by using rules to extract new facts from text, these rules are very simple in form. Reliance on simple relationships has been shown many times in this document to be insufficient for

capturing many of the nuances of natural language. Tellingly, N.E.L.L. also has no ability to decide whether or not its own knowledge is valid. Such metacognition is an essential component of human-level reasoning and learning (Metcalfe & Shimamura, 1994).

7.7 Summation

It is tempting to believe that entities such as Google, Microsoft, or IBM, with the immense linguistic, computational, and human resources at their disposal, will ultimately, perhaps even fairly soon, make truly significant progress on many of the hardest NLP problems. However, since these organizations are deeply committed to relatively knowledge-lean[19] or even statistical approaches to NLP, progress forward from these large-scale projects will be slow at best and non-existent at worst. So far, the main line of evidence that has been presented against the suitability of knowledge-lean approaches for the purpose of performing human-level NLP tasks has taken a very pragmatic path. However, the next section will investigate how some of the theoretical underpinnings of the knowledge-lean NLP methodology are suspect based on first principles. One of the primary indications that these models might be ill-suited for the goals of NLP is that the field has adopted metrical statistics that do not adequately measure performance in a qualitatively meaningful way (i.e. by focusing primarily on local structures, while ignoring the global relationships and the influence of context that are the hallmarks of natural languages. This troubling state of affairs has been recognized as a major flaw of the field by several well-established researchers. Furthermore, alternative metrics have been proposed, but they have not been widely adopted.

Indeed, all of the criticisms above are based primarily on examining the performance of systems that have been created thus far using knowledge-lean techniques. Even though there is the argument that knowledge-lean approaches are unlikely to result in human-level NLP because corporations with essentially unlimited resources and the potential for massive financial gains have been unable to leverage such systems to

[19] The fact that Watson, Siri, and Wolfram|Alpha are capable of compiling information from structured data-bases does not automatically mean that they are "knowledge-rich." Knowledge-richness has as much to do with the internal representations and algorithms employed by a system as does the kinds of knowledge over which such a system operates.

provide services that meet the market's demand for functionality, this does not prove that it is impossible to create those systems using a knowledge-lean methodology. Instead, this only proves that contemporary efforts that do so have failed. In the interests of scientific inquiry, it is then necessary to attempt to examine the mathematics behind individual knowledge-lean technologies to determine whether or not they are inherently unfit for the task of human-level NLP. The nature of this inquiry involves dissecting some of the most widely used knowledge-lean frameworks in fine detail and then seeing if there are common trends among those platforms that place them at odds with what is known about the structure and complexity of natural languages.

Although it may seem pessimistic to dismiss all of these efforts, there is direct evidence from the statistical NLP community that the current mode of operation for the engineering of new NLP system is antithetical to the very fundamental characteristics of the statistical methodology—namely flexibility and robustness. From a lecture delivered at a contemporary M.I.T. course on NLP called "Advanced Natural Language Processing" (Barzilay, Fall 2010), the typical development cycle of statistical NLP platforms is cast as the following sequence of events: 1.) Obtain a corpus, 2.) Build an initial model, 3.) Evaluate the most significant errors in the model and devise manual corrections for them, 4.) Modify the current approach by: a.) Using new features[20] of the data, b.) Modifying the model structurally, and/or c.) Use[21] a more appropriate machine learning formalism, 5.) Repeat steps 3-5 as necessary.

This prescription is clearly targeted at machine-learning techniques, but the process it describes is very similar to the pattern of development of knowledge-rich expert-systems-like approaches. Rather than attempting to make a model that is flexible in a general way, researchers that follow that development cycle will be creating essentially inflexible models that are only flexible at handling a particular, manually identified and designed subset of a more general problem. Since the metrics used by the statistical NLP community are based on very stilted measurements for artificially simple

---

[20] I.e. patterns that can be mined from the corpus.

[21] Or invent, because, in practice, many new formalisms have been motivated to solve very specific types of problems, rather than expending that effort on finding more general solutions. See (Scheffer, Decomain, & Wrobel, 2001) for an example: "Active Hidden Markov Models" are introduced to solve a problem motivated directly by the desire to automatically extract information from semi-unstructured website text.

tasks, it is inevitable that a plan such as the one proposed above would lead to models that are "overfitted" to individual NLP problems and would thus never be usable for general-purpose NLP. The most crippling aspect of this situation, however, is that there appears to either be no awareness that this is the case or that there has been a radical acceptance among the statistical NLP research community that there is no other way to proceed. It would be understandable if this slideshow had instead been labeled "NLP epicycle," where a larger "NLP cycle" was devised to use the lessons learned from error analysis in a much more general way than tweaking individual models for particular NLP tasks, over even worse, for particular NLP corpora for particular NLP tasks[22].

[22] Based on the material covered in the section on the PARSEVAL metric for syntactic parsing, it seems as though many NLP models are simply being optimized to solve the constituent parsing problem for the types of sentences in the Penn Treebank WSJ corpus. While there is bound to be some overlap between the operations required to solve this particular problem and the solution to the problem of general NLP, certainly, the task of parsing the Penn Treebank might not be directly extensible to solving the human-level NLP problem.

8. NLP Metrics

8.1 Introduction

History has demonstrated that the original knowledge-rich, classical AI approaches to NLP were too brittle and inflexible to obtain acceptable results and were rapidly outstripped by statistical methods results. Moreover, from the example NLP tasks described above, it is clear that a certain level of success has been achieved by the computational NLP research community. A wide variety of linguistic tasks can now be performed automatically at what, at least superficially, appears to be commensurate with human-level performance. Furthermore, all of the highest-performing, state-of-the-art approaches given in Chapter 3 could be classified as members of the knowledge-lean, data-driven AI/NLP methodological branch. This lends credence to statements made by proponents of the knowledge-lean approach that it is not only sufficient to capture human-level performance, but also has distinct advantages over knowledge-rich approaches. Some researchers have even searched for evidence to "prove" that the human cognitive system is equivalent to some kind of purely statistical reasoner, usually employing a form of Bayesian inference to make models of human cognition (Narayanan & Jurafsky, 1998). Although it is difficult to compare machine and human performance on individual NLP task metrics, many of the NLP tasks discussed above either report absolute metrics that appear to be extremely good (as in PoS tagging) or that approximate human performance (as in Word Sense Disambiguation). Thus, it is clear why many researchers are optimistic about the progress that has been made in the NLP field.

Given the impressive results obtained by state-of-the-art NLP systems on a wide variety of NLP sub-problems and the optimistic outlook of major researchers in the statistical NLP school, it may seem that there has been significant progress toward automating human-level NLP and that there should be no pressing need to critically analyze the field to highlight its shortcomings. However, the performance of almost all NLP systems is hugely skewed due to two major factors: 1.) the use of metrics that are inappropriate for assessing whether an approach has captured the full-range of complexities of natural language phenomena and 2.) restricting projects to focusing on artificially isolated linguistic phenomena (e.g. part of speech tagging) or by greatly

limiting the scope of problem, as is usually done for reference resolution tasks, in which the types of entities that are considered for analysis are considerably restricted from the range exhibited in human language use. Through performing a new analysis of the NLP state-of-the-art with this information, a different view emerges that reveals the actual extent of the difference between the current NLP approaches and human-level performance. Not only does this realization motivate explorations into alternative approaches to those employed by the presently statistically-dominated NLP field, but also demonstrates that the difference between the current NLP techniques and the human-level of performance is more of a distinction of kind rather than degree.

Another disturbing fact about the purported success of the NLP field is that, by a large margin, the greater proportion of current NLP research is devoted to developing technologies that scale up in ability along with other independently advancing information processing technology—the size of training corpora, the number and speed of computer cores, and the amount of available memory. The metrics discussed below enable NLP researchers, as well as general AI researchers to advantage of this trend of development without significantly adapting the sophistication of the structured knowledge or inference in the computational techniques they use. Aside from the fact that Moore's Law-like trends are now thought to be incapable of perpetually sustaining the historical growth curve for many of these integral resources (Kish, 2002), without developing more effective algorithms, even if such growth was unbounded, it would be highly unlikely lead to the eventual attainment of human-level performance. This is due to the fact that many of the problems in AI, when they are essentially unconstrained[23] and must solved by inefficient algorithms, are so large that no realistic growth of computational resources would be sufficient to cover the sizes of those problem spaces

[23] Recall that Cognitive Science research, as described above, has suggested that the human cognitive system has been subjected to evolutionary pressures to create a time and accuracy-constrained reasoning substrate for solving real-world problems in the domains of physical and social reasoning. The study of the use of conceptual metaphors (Lakoff, The Contemporary Theory of Metaphor, 1993) and analogies (Gentner, 1983) (Hofstadter, 1996) in both linguistic and non-linguistic cognition have provided evidence that humans are able to reason efficiently in other domains because they have the cognitive mechanisms required to map such problems to corresponding representations for physical or social reasoning. Cognitive architectures, such as Polyscheme (Cassimatis N. L., Polyscheme: A Cognitive Architecture for Integrating Multiple Representation and Inference Schemes, 2001), have been developed to utilize this type of reasoning framework.

without the utilization of efficient solution procedures (Dasgupta, Papadimitriou, & Vazirani, 2006).

8.2 F1-Scores

For many NLP sub-problems that involve simple linguistic structures, the most commonly used evaluations are based on a F1-score metric that defines accuracy as a ratio of precision and recall. These measurements are motivated by analogy to the Type I (false positives) and Type II (false negatives) errors that are used in evaluating the success of hypothesis testing techniques for classification tasks (Powers D. M., 2008). In this context, precision is defined as the ratio of true positives to the sum of the true positives and the false positives:

$$P = \frac{True\_Positives}{True\_Positives + False\_Positives}$$

Figure 4. Definition of precision.

Recall is defined as the ratio of true positives to the sum of true positives and false negatives:

$$R = \frac{True\_Positives}{True\_Positives + False\_Negatives}$$

Figure 5. Definition of recall.

The F1-score, which combines precision and recall, is often used as the overall gauge of accuracy for applications of information processing (Makhoul, Kubala, Schwartz, & Weischedel, 1999):

$$F1 = \frac{2 * P * R}{(P + R)}$$

Figure 6. Definition of F1-Score.

The calculated F1 quantity indicates how well a given model avoids Type I and Type II errors.

Some NLP sub-problems produce sufficiently simply structured outputs so that F-scores can be used directly. Notably, Part of Speech Tagging, Named Entity Extraction, and most basic Phrase Chunking techniques all primarily use F-scores to gauge their performance and to provide a method of comparison between different approaches. Typically, whatever approach produces the best F1-score is considered to be the current holder of the title of "state-of-the-art." Care is taken to ensure that the differences between approaches are statistically significant whenever a new method achieves the best published F1-score for a particular NLP task. For recent advances in the field, the absolute increase in F1-scores between subsequent state-of-the-art systems is fairly small; occasionally, approaches differ by only a fraction of a percent. These results are typically taken to establish a strict order of effectiveness among the various formalisms that have been applied to a given problem.

However, it has been argued that performing empirical evaluations of NLP systems in this way is not an entirely sound process. One of the major issues is that published F1-scores tend to be compared directly, despite the fact that the papers associated with each F1-score may have used different corpora and considerable different amounts of training and test data (c.f. (Brill, 1992)[24] and (Leidner, 2003)). For some sub-problems, it may be inappropriate to directly compare certain approaches with others, as they may provide analyses in fundamentally different ways or be subject to significantly different restrictions. These considerations appear to often be ignored by researchers in the field, who are usually concerned with the maximum F1-score that they can obtain through any given experimental setup.

This liberal reliance on statistical measurements of performance is concerning because each split of a specific corpora into training sets and evaluation sets, coupled with whatever internal configuration of a system's parameters is selected, constitutes a unique experiment. While the results of a given experiment may be significant, due in large part to the large amounts of data involved, the performance of NLP systems within the entire experiment-space can vary substantially depending on how a given trial is

[24] Namely the section titled "Results."

constructed. In light of this, it is somewhat disingenuous to claim that an approach is superior to others based on one or even a small number of experiments. Since individual researchers tend to become experts in a limited number of formalisms and therefore are biased towards designing experiments that suggest that their approach is superior to its competitors, which may eventually lead to additional interest or funding for projects based on that particular technique. Additionally, it is not uncommon to find publications that demonstrate that a particular approach, given a more conducive experimental setup, can outperform other competing frameworks that have previously attained a higher published F1-score on a given NLP task.

For some types of approaches, the process of blind modulations of parameters values may produce a wide variety of F1-scores even on the same data set, depending on the values to which those parameters are set. This causes additional difficulty for correctly interpreting F-scores that are reported in the literature. Neural Networks, for instance, exhibit a great variation in their performance that is conditioned on their internal configurations; in the case of Neural Networks, the principal parameters in question are those that define the structure of the network. The number of nodes, connections, hidden layers, whether a recurrent layer is used, and the amount of training examples all significantly impact the performance of connectionist approaches.

Even when F1-scores can be compared appropriately, they are often used to measure performance in ways that do not reveal much about the operation of NLP systems that is applicable to the goal of working towards human-level language use. One of the greatest abuses of the F-Score is to report results based on local structures in a way that obscures the global performance of the system. For the PoS tagging task, the F1-Scores reported traditionally are based on the tagging of individual words, rather than the amount of completely accurately tagged sentences. If the average sentence in a corpus has 20 words, then an approach that can tag 95% of words correctly might commit an error in almost every sentence. Because NLP tasks largely rely on preprocessing—for instance, parsing often requires words to have been tagged with their part of speech—claiming that these sub-problems are more-or-less completely solved by the state-of-the-art is extremely misleading. Phrase chunking uses F-Scores in an almost identical way: reporting only whether individual phrases are correctly chunked, not whether every

phrase in a sentence was chunked correctly. By glossing over the true performance level of NLP systems, the field is lead away from realizing the goal of advancing to human-level sophistication.

As a concrete example of how locally-scoped F1-Scores can greatly differ from measurements of global correctness, consider the task of performing PoS tagging on a fairly complex sentence, which is taken from an online guide released by the Penn Treebank (Marcus, Marcinkiewicz, & Santorini, 1993). For this experiment, the latest version of the Stanford University NLP Group's downloadable parser,[25] will be used as the PoS tagger to be evaluated for accuracy using an F1-Scoring scheme. In order to compute the F1-Score for PoS tagging, it is extremely useful to have an independently created gold standard tagging to evaluate against. For many NLP tasks, especially PoS tagging, there is enough ambiguity and leeway for multiple "human-correct" taggings to exist. Thus, the gold standard will serve as a benchmark of correctness independent of individual preferences. Since the Stanford Tagger is at least partially trained on the Penn Treebank Corpus, the tagging that it produces will be compatible with the Penn Treebank gold standard.

The sentence used in this evaluation is: *Battle-tested Japanese industrial managers here always buck up nervous newcomers with the tale of the first of their countrymen to visit Mexico, a boatload of samurai warriors blown ashore 375 years ago.* The Penn Treebank gold standard tagging is: *Battle-tested*/JJ *Japanese*/JJ *industrial*/JJ *managers*/NNS *here*/RB *always*/RB *buck*/VBP *up*/RP *nervous*/JJ *newcomers*/NNS *with*/IN *the*/DT *tale*/NN *of*/IN *the*/DT *first*/JJ *of*/IN *their*/PP *countrymen*/NNS *to*/TO *visit*/VB *Mexico*/NNP ,/, *a*/DT *boatload*/NN *of*/IN *samurai*/FW *warriors*/NNS *blown*/VBN *ashore*/RB *375*/CD *years*/NNS *ago*/RB ./. The tagging produced by the Stanford Tagger is: *Battle-tested*_JJ *Japanese*_JJ *industrial*_JJ *managers*_NNS *here*_RB *always*_RB *buck*_VB *up*_RP *nervous*_JJ *newcomers*_NNS *with*_IN *the*_DT *tale*_NN *of*_IN *the*_DT *first*_JJ *of*_IN *their*_**PRP$** *countrymen*_NNS *to*_TO *visit*_VB *Mexico*_NNP ,_, *a*_DT *boatload*_NN *of*_IN *samurai*_**NN** *warriors*_NNS *blown*_VBN *ashore*_RB *375*_CD *years*_NNS *ago*_RB ._. Differences (i.e. mistakes) in the Stanford

[25] This parser can be downloaded at: http://www-nlp.stanford.edu/software/stanford-postagger-full-2013-04-04.zip. Version 3.15 is the version used in this evaluation.

tagging are marked in bold. To compute the F1-Score for this sentence, the formulae in Equations 1-3 above can be used. The precision score can be calculated by: $\frac{\#\ Correct\ Tags\ from\ Stanford\ Tagger}{\#\ Correct\ Tags\ from\ Penn\ Treebank} = \frac{30}{32} = 0.9375$. The recall score can be calculated by: $\frac{\#\ Correct\ Tags\ from\ Stanford\ Tagger}{\#\ All\ Tags\ from\ Stanford\ Tagger} = \frac{30}{32} = 0.9375$. Thus, the F1-Score is: $F1 = \frac{2*P*R}{(P+R)} = \frac{2*0.9375*0.9375}{(0.9375+0.9375)} = 0.9375$. Based on an F1-Score evaluation as it is performed by state-of-the-art NLP researchers, the Stanford Tagger achieved an accuracy of 93.75% on this test. While the mistakes made in this sentence are relatively minor, the fact still remains that the entire sentence is not tagged correctly. Therefore, the global correctness accuracy is: 0%. Although this may seem unfairly harsh, even one mistake in PoS Tagging is enough to completely change the meaning of a parse that is based on that tagging.

For an example of how current NLP systems differ from humans, consider that humans do not score perfectly on PoS tagging tasks. In some cases, it is actually impossible to distinguish between different tags for the same word in a sentence (e.g. *The duchess was entertaining* could be understood as having *entertaining* tagged as either an adjective or a participle). Yet, these errors do not interfere with the ability of humans to ultimately comprehend the utterances they encounter. Unlike present-day NLP systems, humans can use context and predictive expectations to recover from errors in determining the role of individual words. In fact, people can still recover a great deal of meaning from sentences that contain words that have previously never been encountered (e.g. the poem Jabberwocky (Carroll L. , 1872) by Lewis Carroll). Furthermore, the state-of-the-art PoS taggers do not have a sentence-level accuracy that is nearly as high as their accuracy on individual tokens. The best PoS taggers produce correct taggings on complete sentences between 55-57% of the time (Manning C. , 2011).

There are other ways that researchers can manipulate statistics to bolster the reported performance of their systems. For instance, for some tasks, both labeled and unlabeled precision and recall scores are reported. In this context, labeled and unlabeled refer to structures that have at least two components: a label that denotes the type of structure and a boundary that defines the extent of the structure in the text. Unlabeled precision and recall only assess the boundaries of structures, not their labels. Therefore, a

sentence could be parsed into entirely incorrect phrases, but if the boundaries of the phrases were correct, this would result in a score of 100% unlabeled accuracy. As an example, consider the sentence *Running costs increase*. The most likely parse of this sentence (according to the standard rules of English grammar) would be to demark *Running costs* as a noun phrase and *increase* as a verb phrase. However, if the parse was based on a tagging of *Running* as a verb, *costs* as a noun, and *increase* as a noun, the final result would be that *Running costs* would be a labeled as a verb phrase and *increase* would be labeled as a noun phrase. A labeled precision metric would score this result with 0% accuracy; an unlabeled precision metric would score it with 100% accuracy.

Despite the already unrealistically optimistic results that are reported by applying F1-scores to NLP sub-problem assessments, some researchers actually criticize this metric for being too harsh. For instance, in the Named Entity Resolution case, it is possible to identify only a portion of a correct named entity, which may happen if, say, the first word of the entity name is not typically used as an adjective, such as *Old Moore's Almanack*, which might be labeled as *Old* [*Moore's Alamnack*]. Even if the identified region is within the boundaries of the correct inference, the overall impact on stages of parsing that rely on these lower-level tasks is great, especially if a pipeline type architecture is used. Mistakes made in an early stage of a pipelined system tend to propagate rapidly.

One final problem with F1-Scores and related metrics is that it prevents researchers from delving deeply into the errors that their systems make. It is more permissible to disregard errors as a product of noise when the system's performance is reported to be near 100%. Many papers do not provide any analysis of errors at all, either in the scope of a particular experimental trial (Collins & Roark, 2004)[26], or more importantly, in the scope of inspecting the inherent deficiencies of the algorithms utilized to solve individual NLP subtasks (Schuemie & Mons, 2005)[27]. Occasionally, researchers will examine the more systematic errors that their approaches make. When this is done, a specific patch

[26] A specific algorithm using Perception-base heuristic search to parse sentences according to predefined features.
[27] In this case, the subtask of Word Sense Disambiguation.

tailored to a few individual classes of mistakes tends to be the deepest amount of analysis performed (Toutanova & Manning, 2000)[28].

8.3 Metrics for Sentence Parsing and PARSEVAL

For more complex linguistic structures, NLP researchers usually do not use a binary (i.e. right or wrong) scoring of the output generated by their systems. Instead, they measure some definition of similarity between what their programs produce and the accepted "gold standards" identified by humans. Gold standards are usually formulated by individual research groups to accompany established corpora that are used to train and evaluate data-driven NLP systems (e.g. a noun-phrase parsing standard for the Penn Treebank corpus (Vadas & Curran, 2007)). Thus, in order to determine the accuracy of, say, the constituent[29] parse of a sentence, it is necessary to compare the results of the parsing algorithm in question to the gold standard parsing solutions that accompany the evaluation corpus.

In constituent sentence parsing, the resulting phrase structure is representable as a tree. Therefore, evaluation of parsing tasks is essentially a comparison of tree structures. Although there are several different metrics for comparing parse trees, one of the easiest to calculate is known as PARSEVAL (Black, 1992). However, PARSEVAL has been widely criticized as a poor metric for capturing the true fitness of a parse.

---

[28] E.g. Looking at capitalization information to improve accuracy on tagging of proper nouns. Even though people do use capitalization as a cue for determining word sense, the use of capitalization as a primary or the only cue for disambiguation is unrealistic. Primarily, capitalization is restricted to the text-based uses of language and humans are clearly capable of understanding proper nouns in speech through other context-based clues. Even within text, capitalization is often ambiguous, as the first word of every sentence is typically capitalized regardless of whether it is proper or common. Texts can also have no capitalization or distorted capitalization, especially when encountered in informal or electronic communications. Considering that a large number of words can be tagged correctly once capitalization is taken into account and that there exist simple, knowledge-lean techniques that can be easily implemented, there seems to be little incentive to invest a large amount of time and effort into producing more sophisticated solutions to this problem. Since the reported performance of modern PoS taggers is already extremely high, the amount of relative improvement that could be garnered by focusing on this one type of error is low, especially since most readers would assume that the baseline accuracy for subtasks is 0% (although, as will be seen below, this is untrue). The ramification of such practices is that more principled, knowledge-rich NLP methods tend not to be developed because individual patch-fixes on top of fairly simple algorithms can outperform most hand-engineered knowledge-rich expert systems, at least when tested on novel corpora. Yet, only such knowledge-rich systems have any possibility in culminating in general human-level performance, since they do not rely solely on the limited types of inference that can be employed by statistical NLP methods.

[29] I.e. as opposed to a dependency parse of a sentence. See Figure 2 in (Nivre, 2005) for an illustration of the differences between constituency and dependency parsing.

To understand why PARSEVAL is far from an optimal metric for NLP parses it is necessary to investigate the actual algorithm in some detail. The primary reason why PARSEVAL is unreliable is that it primarily considers the boundaries of the constituent phrase nodes, as opposed to the lineages of nodes throughout the parse tree. Higher groupings of phrases actually are quite important for determining the fitness of a phrase. Thus, PARSEVAL might cause errors in both possible directions: it can over-value poor parses or undervalue good parses. Although it is important to ensure that highly probable parses are recognized as such, it is perhaps even more important to grade unlikely parses with low scores. The reasoning behind how this can impact the direction of the NLP field is simple—if a popular metric causes a particular approach to appear more successful than others, it is almost a sure thing that that particular approach will be favored over its competitors by researchers in the field. This trend can ultimately cause more promising avenues of research to be abandoned in favor of eking out the most from modeling formalisms that inevitably will prove to be unsatisfactory. Many researchers have disputed whether PARSEVAL actually produces measurements of parse fitness that align with human expectations (Kakkonen & Sutinen, 2006).

Regardless of this subjective judgment, PARSEVAL's performance is highly dependent on the based on the particular bracketing idiosyncrasies of individual corpora (Manning & Schutze, 1999). Consider two alternatives to bracketing the sentence *Hadson Corp. said it expects to report a third-quarter net loss of $17 million to $19 million because of special reserves and continued low natural-gas prices*:

$$
\begin{aligned}
&(\ (S\ (NP-SBJ\ \text{Hadson Corp.})\\
&\quad (VP\ said\\
&\qquad (SBAR\ 0\\
&\qquad\quad (S\ (NP-SBJ-1\ it)\\
&\qquad\qquad (VP\ expects\\
&\qquad\qquad\quad (S\ (NP-SBJ\ *-1)\\
&\qquad\qquad\qquad (VP\ to\\
&\qquad\qquad\qquad\quad (VP\ report\\
&\qquad\qquad\qquad\qquad (NP\ (NP\ a\ third-quarter\ net\ loss)
\end{aligned}
$$

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad (PP\ of \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad (NP\ (QP\ \$\ 17\ million\ to\ \$\ 19\ million) \\
\quad * U *))) \\
\quad (PP - PRP\ because\ of \\
\quad (NP\ (NP\ special\ reserves) \\
\quad\ and \\
\quad\ (NP\ continued\ low\ natural - gas\ prices))))))))))\ .))
\end{array}
$$

Figure 7. This is an example of a sentence as it is annotated in the Wall Street Journal Corpus of the Penn Treebank

$$
\begin{array}{l}
(ROOT \\
\ (S \\
\ \ (NP\ (NNP\ Hadson)\ (NNP\ Corp.)) \\
\ \ (VP\ (VBD\ said) \\
\ \ \ (SBAR \\
\ \ \ \ (S \\
\ \ \ \ \ (NP\ (PRP\ it)) \\
\ \ \ \ \ (VP \\
\ \ \ \ \ \ (VP\ (VBZ\ expects) \\
\ \ \ \ \ \ \ (S \\
\ \ \ \ \ \ \ \ (VP\ (TO\ to) \\
\ \ \ \ \ \ \ \ \ (VP\ (VB\ report) \\
\ \ \ \ \ \ \ \ \ \ (NP \\
\ \ \ \ \ \ \ \ \ \ \ (NP\ (DT\ a)\ (JJ\ third - quarter)\ (JJ\ net)\ (NN\ loss)) \\
\ \ \ \ \ \ \ \ \ \ \ (PP\ (IN\ of) \\
\ \ \ \ \ \ \ \ \ \ \ \ (NP \\
\ \ \ \ \ \ \ \ \ \ \ \ \ (NP \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ (QP\ (\$\ \$)\ (CD\ 17)\ (CD\ million)\ (TO\ to)\ (\$\ \$)\ (CD\ 19)) \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ (NNS\ millionPP - PRP)) \\
\ \ \ \ \ \ \ \ \ \ \ \ \ (PP\ (RB\ because)\ (IN\ of) \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ (NP\ (JJ\ special)\ (NNS\ reserves)))))))))
\end{array}
$$

$(CC\ and)$

$(VP\ (VBD\ continued)$

$(NP\ (JJ\ low)\ (NN\ natural-gas)\ (NNS\ prices)))))))\ (..))$

Figure 8. This is an example of a sentence as it is parsed by the Stanford NLP Group's on-line parser (Stanford University NLP Group, 2012). It is conceivable that a Treebank could be set up to annotate sentences in this way.

Notice that the tree structures in the above bracketing examples are radically different. Namely, the Penn Treebank uses as shallow of a parse tree as possible. This includes grouping together compound nominals into single brackets rather than decomposing them in two their components. For instance, consider the differences in bracketing the nominal *a third-quarter net loss* in Figure 7 versus Figure 8. One of the major weaknesses of the Treebank approach is that the structure of annotations greatly influences the ultimate configuration of statistical parsers that learn from such corpora (Klein & Manning, 2001), which makes the process of comparing models trained on different corpora a more difficult problem (Clark & Curran, 2009).

However, the factor of bracketing methodology also greatly impacts how highly PARSEVAL grades a parse from a model that is trained on such a corpus. In certain cases, the learning of Treebank-specific patterns is so successful that PARSEVAL reports that unlexicalized models can perform nearly as well as lexicalized models (Klein & Manning, 2003). Such a result suggests that PARSEVAL is a bad gauge of parsing effectiveness (Manning & Schutze, 1999). Additionally, PARSEVAL does not work consistently well when applied to parsers in different languages (e.g. German (Rehbein & van Genabith, 2007)). While this may largely be a product of how sensitive PARSEVAL is to idiosyncrasies in Treebank annotation schemes, it may also be attributable to the fact that it is not a very useful overall metric. Ultimately, there should be a decoupling between the evaluation metrics that are used to evaluate models and the Treebanks that are used to train them. It is a difficult process to compare the effectiveness of Treebanks in itself (Kubler, Maier, Rehbein, & Versley, 2008) without having to also determine how well they interact with the various evaluation metrics currently deployed in the field.

Some of the deficiencies of the PARSEVAL metric have been addressed by the introduction of alternative measures of parsing performance. For instance, there are multiple ways to compare the structure of trees. Whereas PARSEVAL compares the boundaries of bracketed "phrases," it is also possible to compare trees by examining their ancestor-descendant relationships. For an example, consider the as "leaf-ancestor assessment" measurement, which purportedly gives a truer sense of the accuracy of a parse than PARSEVAL (Sampson, 2000). Troublingly, PARSEVAL's evaluations are not correlated with the evaluations obtained by ancestor-leaf assessment. This lack of correlation strongly suggests that the two metrics are not actually measuring the same aspect of parser quality. While this does not necessarily indicate that one metric is wrong and the other is right, it raises the question of what is actually being measured in the studies that use these metrics. It may be the case that there are multiple ways to evaluate parse fitness that are somehow uncorrelated. However, this alternative seems dubious from the perspective of Ockham's Razor, as it would imply a more complicated parser is necessary for understanding human-generated natural languages.

Despite this controversy, most alternative metrics are more difficult to implement that PARSEVAL. Accordingly, all of the state-of-the-art approaches currently[30] listed on the ACLWiki use PARSEVAL as the evaluation metric. Other alternatives to PARSEVAL focus on capturing the higher-level grammatical and logical fealty of parse sentences above lower-level syntactic structure matches (Briscoe, Carroll, Graham, & Copestake, 2002). Related to this kind of method are techniques that evaluate the predicate-argument structure of propositions that can be derived from resultant parses (Briscoe & Carroll, 2006). A more in-depth look at how to create, implement, and deploy a more valuable parsing scheme is discussed in: (Carroll, Briscoe, & Sanfilippo, 1998). The authors of that paper point out many potential flaws that a parsing evaluation scheme might exhibit. One of the most compelling observations is that the report that X% of a sentence's structure has been correctly captured reveals nothing about how essential the 100-X% of incorrectly parsed structure is to the eventual understanding of the sentence. Again, these authors advocate for the adoption of a parsing evaluation metric that is based on grammatical relationships instead of syntactic structure. The

[30] Again, "currently" in this dissertation refers to the Spring of 2013.

upshot of such a change in the focus of metrics is that parsers will inevitably need to change the types of features that they attempt to extract from text. If it turns out that parsers are to be evaluated on the semantic interpretations of sentences, then a radical shift in the focus of the field would almost certainly have to follow. On the other hand, the existing large-scale corpora are not intended for anything beyond the parsing of phrase structures or dependency structures.

Attempting to obtain an accurate measurement of the correctness of a parse is important for developing NLP systems that perform at the human-level. Consider that humans do not always parse utterances correctly, at least in their initial attempt, but such failures can be difficult to notice, since humans are good at cordoning off local ambiguities from the rest of the interpretation process (Altmann, 1998) and can use commonsense reasoning with naïve sematic theories (Dahlgren, 1988) to make accurate predictions of how to best reconcile disfluencies based on past experience. None of these features of human parsing are reflected explicitly in the design of knowledge-lean parsers. Furthermore, even if a parsing error results in a communication failure, humans can often use contextual clues, including what may be obtained from interacting with the source of the utterance, either by asking questions of a speaker or re-reading a text. It would therefore be sensible to compare what amounts to a first-pass parse by a one-shot NLP algorithm with whatever the initial human parse of a sentence is, rather than what humans decide is the gold standard after a long period of conscious deliberation.

8.4 Reference Resolution

It is important to reiterate that the primary purpose of this dissertation is not to claim there has been absolutely no progress in the NLP field. Rather, its intent is only to demonstrate that the current standards and practices of NLP research are not conducive to realizing human-level technologies. Not only does the use of misleading statistical measures skew the emergent behavior of research in the field by leading it to favor approaches that provide incremental increases in efficiency over approaches that make significant qualitative progress towards human-level NLP, but it also directly leads to self-imposed limitations by researchers in the NLP field by formulating statistical measure that are compatible with existing approaches. This is a detrimental cycle that

retards progress in the NLP field. However, reporting statistics that are guaranteed to appear impressive is only one of the techniques used to manipulate scientific results. It is also possible to define the parameters of a task so that only the easiest sub-problems are considered. When this occurs, researchers are more likely to develop methods that are tightly fitted to solving these individual problems without much concern for whether such methods can be generalized in the future.

The work done by statistical NLP researchers for reference resolution tasks is probably the best example of how modifying a problem statement to fit existing methods stagnates the advancement of scientific progress. As a prime example, consider the guidelines for the MUC-7 Conference (c.f. (Hirschman, 1997)), which was one of the most recent and more influential summits for researchers interested in reference resolution. Researchers were invited to use MUC-7 corpora marking and evaluation methods to compare the proficiencies of their chosen methodologies on reference resolution tasks. However, the actual definition of what counted as reference resolution was strictly and arbitrarily defined by restrictions on what types of data were considered to be annotatable or "markable" in the corpus.

A large number of potentially significant types of reference resolution situations were listed as unmarkable. For instance, gerund forms can be used to indicate an abstract entity, as in: *Walking is fun*; *it helps clear my mind*. In this sentence, *Walking* and *it* act as co-referents. However, the MUC-7 guidelines prevent such references from being marked in corpora. Structures that are more complex than a noun phrase also cannot be marked. Entire sentences can be embedded within sentences and then referenced: *She asked him if he wanted to leave; he said that it didn't matter*. Co-references to substructures of complex entities are also barred. As an example, consider mapping "it" to the year of a MM/DD/YYYY. Additionally, conjoined or multiple head noun phrases cannot be marked (e.g. A and B went to the store. They did not buy anything). Bare noun mentions, such as the use of *days* in: *Days in winter can be cool* or *Some days are 20% cooler than average*, also not markable.

Although it may seem as though these guidelines are somewhat arbitrary, they actually are formulated to exclude reference resolution problems that would require world knowledge, commonsense reasoning, and other inherently knowledge-rich

techniques. A detailed treatment of the necessity of knowledge-rich approaches for solving general reference resolution problems can be found in: (McShane, 2009). Furthermore, the failure to incorporate world-knowledge actually makes statistical NLP procedures extremely brittle, which is especially disappointing in that handling brittleness was one of the major motivations for adopting statistical methodologies in the first place. An example of this is the matching of pronouns to nouns by gender. Occasionally, news articles (or other sources) will contain an incorrectly gendered pronoun, such as *The suspect was described as a 6' 3"* ***male****;* ***she*** *was last seen running towards Hoosick Street*. Many reference resolution programs are hard-coded to rigidly obey gender agreement. While this is a useful assumption for error-free text, it clearly fails in general. Such scenarios underline the vital importance of knowledge even in what would normally be considered to be a knowledge-lean environment. Yet, it will not be a simple task to create the resources required for a machine-learning approach to knowledge-rich reference resolution. Recall that the corpus-based approach to NLP is essentially incompatible with directly combining knowledge-rich and statistical methods because it is prohibitively difficult, if not impossible, to produce large-scale, knowledge-rich annotations for corpora.

Even if the field moves beyond the limitations posed by the MUC-7 formalisms, this shift will have a significant lag time in its full effect on the literature and on newly developed NLP systems. This trend is caused by the fact that NLP formalisms, when cited in later papers, are usually quoted only in terms of their accuracies and not with a description of their system assumptions and limitations. Although the particular data set, such as MUC-6 or MUC-7, used to produce a particular experimental result may be mentioned, there is usually little or no discussion about the guidelines used to create those data sets. Therefore, papers that quote the accuracy of certain NLP-subtasks are likely to refer to outdated results that are removed from the initial context that defined the metrics used to evaluate the model and the specific restriction on the task definition itself. Thus, even if more realistic corpora can somehow be produced, the systems that process such data will in a sense be competing with formalisms that were scored on what amounts to be a significantly easier challenge. There have been some alternatives to the MUC-7 evaluations that appear to be moving in the correct direction of expanding the

scope of reference resolution to contain more knowledge-sensitive cases (Schlangen, Baumann, & Atterer, 2009). Such metrics aim to be able to accommodate accurate reference resolution results even when grammatical errors and disfluencies are present.

8.5 Machine Translation (MT)

Although the details of machine translation algorithms and the specific concerns relevant to that area of NLP research are beyond the scope of this dissertation, it is nevertheless useful to examine the metrics that are deployed by MT researchers, because this evaluation process is a difficult problem in its own right: an accurate assessment of the fitness of a given translation can only be obtained if some deep understanding of the meaning of the texts involved is taken into account. Early metrics for machine translation, such as BLEU (Papineni, Roukos, Ward, & Zhu, 2002), relied heavily on the exact translation of aligned texts. Text alignment is the process of reorganizing the components of utterances in the source language document with their counterparts in the produced target language document. This type of assessment usually resulted in very low scores because "near-miss" translation errors were not well-tolerated. Newer metrics are based on conceptual entailment (Schaffner, 2004), which seek to determine if the semantic interpretation of a translated document is compatible with the intent of the original source. However, this type of comparison requires a great deal of world knowledge.

8.6 Beyond Individual Metrics

Another potential source of confusion from the way that NLP metrics are reported is that they are typically not compared to the baseline accuracies of extremely simple algorithms. For instance, for the task of Word Sense Disambiguation, the simplest non-trivial algorithm[31] is to generate a lookup table that maps every word in the training corpus to its most common part-of-speech according to some statistical analysis of the language used in the text (Ide & Veronis, 1998). During evaluation, each word in is looked up in the table that was generated during the learning phase. If there is an entry for the given word, then it is tagged with the associated part-of-speech from the table.

[31] E.g. Tagging all words randomly or with the same tag.

Otherwise, the word is tagged as a common noun. For English, this algorithm results in an accuracy of approximately 80%. State-of-the-art approaches therefore do not improve upon this baseline to the extent that would be suggested by only examining the reported results. Many other NLP subtasks exhibit similar trends. For instance, the correctness rate of state-of-the-art word sense disambiguation systems is approximately the same as simply picking the most commonly used sense of each word. Both procedures have an accuracy of about 66%.

8.7 Summation

The field of statistics provides many invaluable tools for all scientific fields. Without empirical methods and mathematically rigorous techniques to evaluate them, it would be impossible to conduct the types of analytic experiments that are required to synthesize general theories. However, it is also possible for misleading applications of statistics to mislead those who are not intimately familiar with why a particular metric is used over another. While it is the case that statistical methods in AI are well-grounded in a mathematical sense, they are not motivated at all from a linguistic or cognitive perspective. Although there is a possibility that local measures of performance might provide some insight into larger scale levels of performance, this is unlikely to be the case for NLP systems, as the few reported statistics that look at the scale of sentences, as opposed to words, show that knowledge-lean approaches are not very successful at processing global patterns. To address the question of whether or not there are reasons to doubt the encouraging predictions that statistical approaches alone will eventually solve the hard problems of NLP and AI in general, one must examine the trends that result from the use of such approaches at the largest scale possible.

9. Statistical, Knowledge-Lean Approaches to NLP

There are a huge number of distinct knowledge-lean approaches that have been developed for various AI applications and most of these either have been or could be leveraged for dedicated NLP work. This diversity is due in part to the fact that many NLP tasks can be cast as a classification problem (Medlock, 2008).[32] Because there are so many methods and because many of these methods are actually deeply related, if not equivalent to each other under reasonable sets of assumptions, it is not feasible to examine or even mention every possible method. For instance, there will be little discussion of the knowledge-lean approaches to reasoning that have not been popularly adopted by the NLP community. It must be assumed that there is some kind of "Darwinian" force at work that, at least in part, guides the trends of this research field. Indeed, almost all of the knowledge-lean methods have been published in multiple journals in multiple subjects with multiple iterations of refinements. Thus, the very fact that a method is not selected to become the basis of NLP models suggests that there is little expected utility of developing such a platform. In any case, it is also evident that even diverse types of underlying models can produce roughly the same level of performance on a task. Consider the following table of data that has been simplified and adapted from the ACLWiki page on the state-of-the-art of PoS tagging performance.

Table 2. Types of knowledge-lean approach with the accuracy they can obtain on a PoS tagging task.

System Description	Accuracy
Trigram Hidden Markov Model	96.46%
MEMM with external lexical information	96.96%
Maximum entropy cyclic dependency network	97.05%
Averaged Perception discriminative sequence model	97.11%
Maximum entropy bidirectional easiest-first inference	97.15%

[32] Examples of this principle are as follows: PoS tagging can be seen as a classification of words to discrete word-sense categories, parsing can be seen as a classification of a sentence into a corresponding general phrase tree structure, reference resolution can be seen as a mapping of mentions to classes of actual entities, and Word Sense Disambiguation can be seen as a classification of meaning along the lines of discrete classes of definitions.

SVM-based tagger and tagger generator	97.16%
Perceptron based training with look-ahead	97.22%
Averaged Perceptron	97.23%
Averaged Perceptron	97.44%
Maximum entropy cyclic dependency network	97.24%
Maximum entropy cyclic dependency network	97.29%
Maximum entropy cyclic dependency network	97.32%
Bidirectional perceptron learning	97.33%
Semi-supervised condensed nearest neighbor	97.50%

In addition, for each of the methods that are mentioned below, a selective amount of detail is provided—namely just enough to demonstrate their operating principles and to derive some common characteristics of their effectiveness in working towards solving the human-level NLP task. The section on the Hidden Markov Model family of approaches is by far the most detailed—this has been done in order to enable those unfamiliar with the content of these models to understand how they can be adapted to study natural language processing. The other intensive sections pertain to Maximum Entropy (ME) Modeling and Probabilistic Context-Free Grammars (PCFGs). Although several other approaches are explored, HMMs, ME, and PCFGs are a representative cross-section of the level of technology available to the NLP field.

9.1 A Brief Discussion of Stochastic Modeling as Applied to the NLP Problem

The intent to use mathematical models to study the properties of natural languages has existed from nearly the onset of the inception of such models. Andrey Markov, a pioneer in the stochastic modeling arena, imagined as early as 1913 that a mathematical technique, which would come to be known as a Markov Chain, could be used to model the patterns of language (Weber, 2011). Markov deployed his model to determine the characteristics of the distribution of the frequency of consonants and vowels in a work of Russian literature (Markov, 1913). Throughout the twentieth century, mathematicians and scientists explored the applications of Markov Chains; by 1975, Hidden Markov

Model-based computers programs were implemented to model the patterns of human speech (Jelinek, Bahl, & Mercer, 1975).

Other approaches to applying stochastic modeling techniques derived from the field of information theory. Claude Shannon, an instrumental figure in the development of the field of information science, derived a formula for the level of entropy of a communication signal (Shannon, 2001). Soon following this formulation, a scientist named E. T. Jaynes noted in two papers (Jaynes, 1957) (Jaynes, 1957) that Shannon's equation was analogous in form to the notion of entropy that Gibbs developed for the field of statistical mechanics (Gibbs, 1928)[33]. Jaynes' contribution was to formulate the "theory of maximum entropy," which held that the statistical model that is the truest to the distribution being modeled is the one that exhibits the most information entropy. For the purposes of statistical mechanics, entropy refers to the level of disorder present in a physical system. On the other hand, the concept entropy in communication can be seen as a measure of the information content of a signal. The more disordered a signal, the more information the signal can convey. Although this may appear counterintuitive, consider that the information in a signal could be represented by a function. As the information increases in complexity, the function required to characterize it must also increase in complexity. Since a more ordered communication signal (i.e. one with low entropy) is highly predictable, a relatively simple function can be used to predict the future behavior of the signal given some previous data.

Even though the intent to develop statistical treatments of natural language was in place by the advent of formal and computation linguistics, not all researchers involved with theoretical linguistics have embraced the notion that natural languages are subject to statistical analysis. Chomsky famously stated that "it must be recognized that the notion of a 'probability of a sentence' is an entirely useless one, under any interpretation of this term" (Chomsky, 1969). For a long time, Chomsky's views influenced the NLP field to such an extent that probabilistic models of language were not taken seriously until empirical results proved otherwise, following the ability to create large stochastic models

[33] Gibb's distribution (a.k.a. the Gibbs Measure) is a re-expressed form of the Boltzmann distribution (Boltzmann, 1872).

of language using computers. The advancement of computing technology has been a major factor in the advent of statistical language modeling.

The accuracy of Chomsky's statement has been a point of bitter contention between different schools of thought in the field of linguistics, although it is now reasonable to claim that most linguists subscribe to the notion that it is both meaningful and fruitful to use statistical information theory to study the human use of language. To conclusively demonstrate that Chomsky's viewpoint is spurious at best, several researchers have used his equally famous "colorless green ideas sleep furiously" example as a strawman to knock down his claims in general. Indeed, Norvig has found evidence that nearly all of the substrings of Chomsky's sentence have, in fact, been published in English (Norvig P. , 2011).[34] There is little doubt that probabilistic methods are useful for studying many aspects of language and that doing so has resulted in significant strides in the field of computational linguistics. Yet, one major negative impact of these findings is that they have led to a climate of research in which the superiority of statistical approaches to NLP is taken for granted. This trend has continued in the field today and has contributed to the decline of knowledge-rich approaches in favor of statistical models.

After Chomsky's initial statements had been counteracted by actual progress made via the deployment of statistical language models, this approach gained a foothold in all areas of NLP. A documentation of how the modeling culture shifted from knowledge-rich to knowledge-lean frameworks is provided in a presentation delivered by Mark Steedman at the Informatics Hamming Seminar held in October 2010 in Edinburgh, Scotland (Steedman, 2010). This presentation examines this aspect of the development of the NLP field, with a particular focus on the application of Automatic Machine Translation (AMT). Like the NLP field as a whole, the pendulum between rule-based and statistical techniques in AMT research has swung back and forth. After initial attempts to use finite-state, syntactically robust, rule-based parsing techniques in

[34] This observation, notwithstanding the vanishingly small probability that an arbitrary speaker of English would both observe and subsequently remember all of these substrings and then go on to apply this information under some statistical framework to evaluate the likelihood of that sentence, is taken to criticize Chomsky's entire program of casting linguistics as something beyond the reach of simple stochastic modeling.

conjunction with semantically-rich information failed to satisfactorily address the issues encountered in AMT research, knowledge-lean models quickly grew in popularity. One of the main reasons for the abandonment of the rule-based approach to AMT was the poor tractability of the algorithms developed to contend with the complexity of the grammatical representations that it employed.

9.2 Stochastic Processes and Associated Models

A stochastic process is a process in which the states of the process change over time in a probabilistic, not deterministic, fashion. The appearance of non-determinism can either be due to a genuinely random element in a process or due to incomplete knowledge about a process's activity, perhaps due to imperfect observations or the inability to enumerate all of the factors that influence the progression of states within a process. Formally, a stochastic process is defined as a set of random variables, X, each of which can achieve some discrete value, labeled x_i. These X represent the possible states that a process can achieve; each the x_i values represent one of the defined process states. The set of possible values for the members of X is designated by S, the "state space" of the process. Because a process unfolds over time, there must be an order imposed on the states that the process achieves. This order is provided by an ordered set of indices, T, such that each member of T, t_i, is a time point during the operation of the process. In this framework, processes proceed over time as a sequence of state transitions, where the states are drawn from X and indexed from t_0 to t_n, where n is the number of time steps during which the model of the process is executed. Each state has a particular probability of transitioning to another state during the next time step. In general, these transitional probabilities may also vary over time.

To create a model of a stochastic process, some structure must be imposed to determine the patterns in which state transitions can occur. In order to fit a model to the behavior of a particular process, it is necessary for the model to have adjustable parameters that influence the probabilities of the possible state transitions. Although the structure of a model and the associated parameters can be sometimes be established directly through expert knowledge, it is usually necessary to train a model automatically by fitting the parameters of the model to a selection of observed instances of a process's

behavior. This training is often performed by inductive machine-learning algorithms. The complexity of a model can be gauged by the number of parameters that it specifies. As model complexity increases, an increasing amount of data is required to properly train the model's parameters, so that a representative sample of each type of transition within the process can be considered. Models provide a mathematical description that allows a simulation of the behavior of a process to be conducted. Conducting such a simulation is often referred to as "running the model."

9.3 Defining and Justifying the Scope of Modeling Techniques Covered

There are far too many variations of machine-learning techniques to cover in a broad assessment of the field. The purpose in providing a review here is to identify the most promising methods that have been developed for NLP applications and then discussing how each has failed. Well-motivated criteria exist for determining which types of formalisms do not need to be included for this endeavor to succeed. As Steven Pinker points out in (Pinker, 1994), languages are best cast as discrete combinatorial systems. Therefore, only discrete models and their associated algorithms will be thoroughly discussed here. Some further information about continuous Markov Chains and Models can be found in (Liggett, 2010).

In addition, there are a large number of variations on and advances to machine-learning-based modeling techniques that do not appear to have yet been purposed for creating NLP models. Although these technologies are undoubtedly significant, they are also outside of the scope of this overview. Also, note that while the difficulties of speech perception (c.f. (Pisoni & Remez, 2005)) or visual perception of written material (c.f. (Rayner, 1995)) are certainly significant and are an important topic for cognitive science, some of these difficulties can be cleanly separated and abstracted away from the issues that arise in the processing of post-perception representations of language. This idealized treatment is the basis for the vast majority of NLP-driven research beyond the specialized field of speech recognition. Technologies and formalisms that are specifically designed to address the processes of sensory transduction are therefore not covered in depth by this assessment of the state of the NLP field.

In consideration of whether the advances and modifications of the basic NLP modeling frameworks can be reasonably expected to lead to the emergence of human-level performance, it is sufficient to say that heretofore, no such algorithm or formalism has been produced that can address the fundamental problems with the current state of statistical NLP modeling. Where the specific objections and criticisms that are outlined in this paper have been addressed by specific advancements, a mention of this will be made; nevertheless, there are issues that transcend all of these concerns, which are centered on the trend of attempting to utilize knowledge-poor systems to solve what is essentially a knowledge-rich task.

10. Markov Models: Markov Processes, Markov Chains, and the Markov Assumption

There are many formulations of stochastic processes, which can be distinguished by some property of the assumptions regarding the random variable and time-dependent aspects of the process structure. The Markov Process (MP) is probably the most relevant to the field of NLP due to the fact that it is a fundamental stepping stone towards the family of Hidden Markov Models that are commonly used for modeling aspects of natural language. MPs that involve discrete state values and are indexed by discrete time steps are sometimes called Markov Chains. Continuous-time Markov Processes have also been studied extensively (Archambeau & Opper, 2011), but Note that the topology of a graphical representation of a Markov Chain does not have to be linear. The most important property of MPs that distinguish them from other formulations of stochastic processes is the fact that the transitional probability between states is dependent entirely on the previous state. This can be formalized by the following expression, using the notation for describing stochastic processes introduced above:

$$Pr(X_i = x_i \mid X_{i-1} = x_{i-1} \ldots X_0 = x_0) = Pr(X_n = x_n \mid X_{n-1} = x_{n-1})$$

Figure 9. The Markov Assumption.

It can thus be said that a MP is memoryless—each state is aware only of the incoming transition and is not conditioned on the activity of the process before the last time step. Models of processes that obey the Markov Assumption are called Markov Models. Although the Markov Assumption is unrealistic for most real-world phenomena, it greatly reduces the number of parameters required to model any particular process. By reducing the number of parameters in this way, Markov Models are much more computationally tractable to train. This degree of tractability, along with the simplicity of the underlying algorithms for utilizing Markov Models, greatly contributes to the popularity and continued use of this framework. An additional assumption made for Markov Models is that the transitional probabilities between the states of a process are time invariant. This assumption also greatly reduces the number of parameters required to fit a Markov Model to a particular stochastic process. With this formulation, the

following parameters are required for every Markov Model: a set of states (usually given by the modeler and not directly learned), a probability distribution that describes the initial state of the process, and a matrix of transition probabilities between each of the states. For a model with n states, there would be $n + n^2$ learnable parameters to fit: n from the probabilities of a process initializing in any given state, and n^2 from the transitional probabilities between each of the states. Only pair-wise transitions need to be learned due to the Markov Assumption; the time-invariance condition ensures that the values of the transitions will not change however long a model is run.

10.1 Visible Markov Models, Finite State Automata, and Formal Languages

If the states of the process and their transitions are fully observable, then the model is said to be a Visible Markov Model. Typically, this condition of perfect observability is never met in practice, but it is useful to consider in illustrating the basic mechanics that underpin all varieties of Markov Models. Almost all approaches to NLP treat natural language as though it can be modeled as a mathematically discrete phenomenon in both time and state space.

Formally, a discrete Visible Markov Model can be defined with the following notation, which is borrowed from (Rabiner, 1989): for a Visible Markov Model, M, let there be a state space, S, that is composed of N states, $\{S_1, \dots, S_n\}$. The state space represents all of the possible states of the stochastic process that is being modeled. If the model is for language processing, then these states might correspond to the individual words of a language. Between each of the N states of the model, there is a probability of transition—note that one possibility is that a state will "transition" to itself. These transitional probabilities can be thought of as an $N\ x\ N$ matrix, denoted by A, in which each element of A, a_{ij}, is the probability that state i will lead to a transition to state j. Additionally, there is a set of indices, T, such that all members of T, t_i, represent consecutive time steps of the process being modeled. At each t_i, the model is in exactly one of the N states. The probability that the modeled process's behavior should follow a given sequence of state transitions is computed by multiplying the transitional probabilities encountered during each transition.

For some models, there might be constraints imposed that make particular transitions impossible. Models in which it is possible to reach each state from every other state by some path of transitions are said to be ergodic. Since the inception of the field of information science, ergodic Markov Chains have been used as a tool for modeling text-based natural language communication (Shannon, 1951). In practice, most Markov Models assume that the indexed time steps have a constant duration; it is equally assumed that the transition matrices of Markov Models are time invariant. Both of these considerations, which vary in degrees of realism depending on the particular process being modeled, are ways to use assumptions to reduce the computational burden of creating a stochastic model of a very complex phenomenon.

As a concrete example, consider modeling patterns in a simple language that only consists of four words: *the*, *cat*, *rat*, *saw*. There are four grammatical sentences in this language: *the cat saw the rat*, *the rat saw the cat*, *the rat the cat saw*, *the cat the rat saw*. Assume that these sentences are equally likely to occur within the language. It is possible to model the generation of sentences in this language as a Markov Process. Each of the words can be considered to be a "state" in the process that generates the language. Let X consist of four variables, one corresponding to each word in the language: THE, RAT, CAT, SAW. The structure of this model can be described as a graph, where the nodes of the graph correspond to the four states of the language. Additionally, for the purposes of this illustration, two special states are introduced: an initial state and an end state. The initial state is unique for these models, whereas there could be multiple end states; here there happens to be a unique end state that corresponds to the end of a sentence.

Directed links between these nodes correspond to possible transitions between the states of the language. These links can be weighted between with a real value between 0 and 1.0, which corresponds to the probability of a transition between the nodes in the direction indicated by the link. The initial state of a model has only outgoing links; the final states of a model only have incoming links. Links between the initial state and other states of the model correspond to the probability distribution parameter of start states for the model. In the language above, all sentences must start with the, so the corresponding graph would have one link from the initial node to the THE node with a weight of 1.0.

For all nodes other than the end states, the outgoing transitional probabilities must sum to 1.0.

Nodes that are independent of each other are typically depicted as having no link between them, as that particular transition is considered to be impossible by the model. This assumption is realistic only in the visible modeling situation, because it relies on the assumption that all possible states and state transitions have been observed, at their proper frequencies, in the data that are used to train such models. Techniques for "smoothing" the transitional probabilities in a model have been developed to account for this potential for error in the non-visible case. The basic representation of a Markov Model is an $N\ x\ N$ matrix whose rows and columns are labeled by the states in the model state space and whose cells are filled by the transition probabilities between the state labeled by the in the row of the cell and the state labeled by the column of the cell. For the language described above, the following matrix represents the corresponding Markov Model:

	Δ	THE	CAT	RAT	SAW	$\square$
Δ	0.0	0.0	1.0	0.0	0.0	0.0
THE	0.0	0.0	0.5	0.5	0.0	0.0
CAT	0.0	0.25	0.0	0.0	0.5	0.25
RAT	0.0	0.25	0.0	0.0	0.5	0.25
SAW	0.0	0.5	0.0	0.0	0.0	0.5
$\square$	0.0	0.0	0.0	0.0	0.0	0.0

Figure 10. Transition matrix for a Markov Model applied to NLP.

Markov Models are also expressible as a graph. The following image is a representation of the graphical form of the Visible Markov Model for the language introduced above. In this image, the initial state is the node that contains a square; the final state is the node that contains a triangle; the states that correspond to the words of the language contain the text of the words they represent:

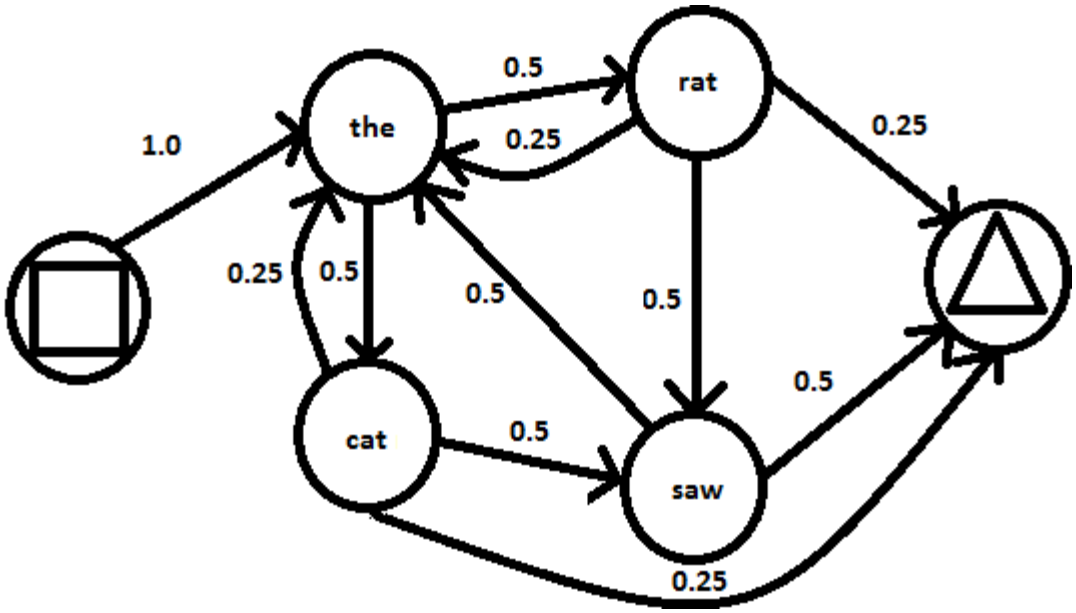

Figure 11. Graphical representation for a Markov Model applied to NLP.

For the purpose of modeling actual natural languages, it is necessary to use a dedicated inductive learning algorithm to compute the transitional probabilities in a Markov Model. However, the limited nature of this toy language enables one to compute the transitional probabilities by hand. An analysis of the sentences of this language reveals that all of them begin with "the." Therefore, there is only one outgoing link from the initial state to the THE state. This link is given a probability of 1.0. Determining the other probabilities is slightly more complex. In two of the four sentences, *the* is followed by *rat*. In the other two sentences, *the* is followed by *cat*. Therefore, the THE node has two outgoing links, one link to the CAT node and one link to the RAT node, each with a probability of 0.5. It is simple to compute the other transitional probabilities through this process.

A model such as the one portrayed above can be used to perform several investigations into the process under study. In this case, two possible uses are: 1.) to determine how likely a given sentence would be generated according to the mechanics of the language being modeled and 2.) to generate a sentence that is similar to the sentences that have been observed as belonging to that language. These two uses provide an illustration of a distinction between the capabilities of the different statistical modeling

paradigms. Along this dimension, there are two broad classes of statistical models: discriminative models and generative models.

Discriminative models can perform the first task listed above; these models can discriminate whether or not a given sequence of observation was likely to have been generated by a particular type of process. To do this, the model must at least take into consideration the conditional probability of state of the process conditioned on the observations of the process utilized for training the model. If Y represents an unobserved state and X represents observed states, then this probability distribution can be expressed as: $Pr(Y|X)$. In the Visible case, the unobserved state is simple— whether or not a sequence was generated by the modeled process; the non-Visible, or Hidden, case can be more complex. In the first example use case above, X is a given sentence and Y is whether or not that sentence is part of the language.

Although discriminative models are valuable tools for a variety of tasks, they are not as powerful as the class of generative models. A generative model is capable of solving classification problem, as discriminative models case; however, only generative models can solve problem similar to the second use case described above. Generative models model the full joint distribution between the unobserved states and the observed states. With the same notation for X and Y used above, this probability distribution can be expressed as: $Pr(X, Y)$. This enables generative models to model how the unobserved states generate the observed states. Markov Models are generative models and therefore are capable of solving both classes of problems stated above.

Using the above model to solve the two sample problems is straightforward and provides some insight into why it might indeed be sensible to speak of the probability of a sentence, contrary to Chomsky's famous position on the matter. To determine the probability that a sentence was generated by the language, it is only necessary to use the words of the sentence as a guide for making transitions in the Markov Model. Starting in the initial state of the model, each subsequent word of the sentence determines the subsequent state of the model. Since the states in this Markov Model are unique, there can be either zero or one matching states for the next observation. In the case where multiple transition choices are possible, the model must sum over the probabilities from each unique path to determine the total probability. If the sentence under consideration is

the cat saw the rat, then the model would transition between the states in the following order: $INITIAL\ STATE, THE, CAT, SAW, THE, RAT, FINAL\ STATE$. To calculate the probability that this sentence is generated by the language, the probabilities of each transition are multiplied by each other. Thus, the computed probability is: $1.0\ X\ 0.5\ x\ 0.5\ x\ 0.5\ x\ 0.5\ x\ 0.25\ =\ 0.015625$.

Because this is a fully visible Markov Model, the state transition path is unique for each unique observation sequence. It will be seen that this is not the case for Hidden Markov Models, in which it is often necessary to sum over several possible paths. On the other hand, to generate a sentence from the model, one begins in the start state, which just defines a probability distribution over the possible initial states of the language, and then selects subsequent transitions between states that have a non-zero probability. For instance, either of the following sequence of transitions could be generated: 1.) $START \rightarrow THE \rightarrow CAT \rightarrow SAW \rightarrow THE \rightarrow RAT \rightarrow END$ 2.) $START \rightarrow THE \rightarrow CAT \rightarrow SAW \rightarrow THE \rightarrow RAT \rightarrow SAW \rightarrow THE \rightarrow CAT \rightarrow END$. Note that the second sequence of transitions would necessarily have a lower probability than the first, since it contains more transitions.

It is worthwhile here to briefly consider some additional metaphors for the operation of Markov Models: those of message passing or the flow of probability. Message passing constructions for algorithms are often used in conjunction with graphical models. Alternatively, inference about the states of a Markov Model can be conceptualized as a flow problem. In this scheme, probability can be conceived of as a mass that flows through a graphical model. At the onset, there is an available probability mass of 1.0, which can only decrease (or remain the same in the event that a state with only one outgoing transition is reached) with at each subsequent time step. This mass is attenuated by the transitional probabilities that it "passes" through, as though the probability defined the volume of flow admitted from the previous state to the next state.

While generative models are able to solve problems that discriminative models cannot, it is typically more difficult to construct a generative model of a process than a corresponding discriminative model of that same process. Some tasks, such as ones that rely purely on classification or tagging of sequences, can be performed by either type of model. For these tasks, it may be beneficial to use a discriminative modeling paradigm

rather than a generative one for a number of reasons. The principal cost of deploying a generative model is involved in having to train the model on data that is sufficient to capture the full joint distribution of observable and unobservable states. Learning the full joint distribution requires learning algorithms with greater computational complexity, which will be further described in the section on training Hidden Markov Models. These algorithms require a large amount of time and memory and so they rely on modeling frameworks to provide simplifying assumptions, such as the Markov Assumption for Markov Models, to run with a feasible amount of resources.

Additionally, the features of the training data that are used to actually train generative models can only be as complex as the structure of the generative model itself. With no theory about the internal mechanisms of a process, the only features of the data that can be used are how the unobserved states of the model vary directly with the observed states. This restricts the example Markov Model above to considering only whether a given sentence entirely belongs to the language or not. In order to make use of more sophisticated features of the data, such as the probability that a sentence is in the language if it contains the word "the," some additional structure would need to be added to the model to account for how the actual process of generating the language itself selects the word "the" in creating its sentences. For many interesting natural processes, including natural language production, it is either very difficult or impossible to produce fined grained theories of the internal mechanisms of a process. Linguists have compiled a number of detailed theories about the production of natural languages and some attempts have been made to produce models that have the complexity of structure required to capture the functions and relationships that are proposed as the underpinnings of how the mind creates language. One example of this work is the PCFG framework developed by Ivan Sag and his colleagues at Stanford University. Although there are systems that implement components of PCFG to solve natural language tasks, it turns out to be very difficult to provide adequately complex tags in large-scale corpora that can capture the information required to learn models based on a framework as detailed as PCFG.

Representing a Markov Model as a graph does more than provide a convenient metaphor for understanding how such a model functions. The ability to construct such a

corresponding graph for every Markov Model demonstrates that they are a form of probabilistic graphical models, a family of models that includes Bayesian Networks. In particular, Markov Models are a special case of Dynamic Bayesian Networks (DBN). DBN's have fewer topological restrictions than Markov Models, which allows them to overcome some of the problems with the Markov framework that will be illustrated in the section on Hidden Markov Models below.

Depicting a Markov Model in graphical form also reveals that Markov Models are equivalent to Non-Deterministic Finite State Automata (NDFSA). There are a number of results that demonstrate that the expressive power of an NDFSA is equivalent to other classes of models that on the outset appear to have less expressive power. For instance, NDFSAs can be converted to Deterministic Finite State Automata (FSA). Methods exist for converting a Markov Model directly into finite-state transducer form (Kempe, 1997). This equivalence is unsurprising due to the fundamental topological similarity between Markov Models and FSAs, as FSAs are themselves a set of states and labeled transitions between states. However, what is more pertinent is that this equivalence demonstrates that statistical approaches are reducible to the types of knowledge-rich approaches that the statistical approaches are purported to supplant. While statistical formalisms may appear to provide greater flexibility, as well as more computationally efficient representations and algorithms, they do not differ much from the knowledge-poor forms of logical models that are no longer considered to be state-of-the-art. Pure statistical models at best exploit the properties of limiting assumptions, such as the Markov Assumption, to improve performance in narrow bands, according to contrived metrics.

An FSA equivalent to the Markov Chain depicted above would appear as shown in the image below. Note that the same convention for representing the initial and end states is used:

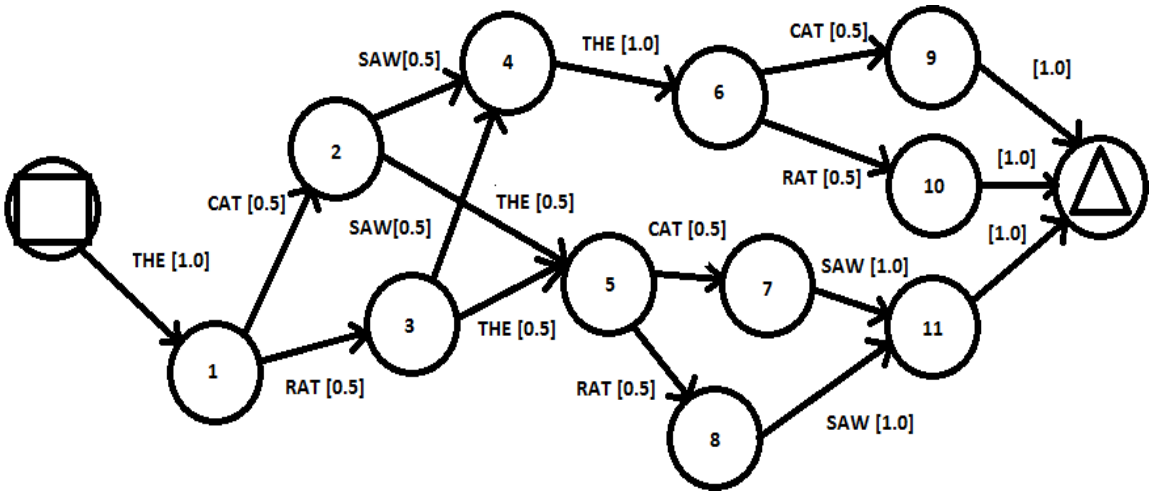

Figure 12. FSA equivalent to a Markov Model.

There are some clear similarities between the Markov Model and the FSA; namely that the probabilities on the outgoing edges of each node (except for the nodes corresponding to the final states) all sum to 1.0. The operation of interpreting or "decoding" the above model is essentially identical to the one for the Markov Model, except that the words of the language are "consumed" by each transition (if the FSA is being used in a discriminative way) or are "produced" by each transition (if the FSA is being used in a generative way). Although the FSA clearly limits the maximum length of sentences to four words, it should be noted that in the Markov Model, any sentence with more than four words would have a lower overall probability than a shorter sentence because each transition reduces the entire probability of the generated sequence, as will be shown in the section below detailing the decoding problem for HMMs.

All finite-state formalisms can be conceptualized as a series of rule-sequences in which the application of the rules is non-recursive. This limitation is due to the fact that FSA-class representations have a limited amount of memory; namely the amount of memory of an FSA is limited by the number of states in the automata. Conceptually, this is caused by the fact that no information about the path taken to a particular state can be stored by the model, so information about the past activity is lost after every transition. The only way to increase the memory of such models is to directly increase the model size by adding more nodes, which creates more parameters to train, which in-turn decreases the tractability of applying these methods to full-scale real-world problems. It

is the ability to store paths (and the corresponding memory capacity) that enables more complex, recursive applications of rules. FSAs are mathematical models of a class of formal languages known as Regular Expressions (RE). A total description of the properties of RE languages can be found in (Hopcroft, Motwani, & Ullman, 1979).

There are a number of extensions to FSAs that ultimately do not increase the expressive power of the class of languages that they represent. Indeed, formal languages are strictly ordered by their expressive power in the Chomsky Hierarchy. RE languages are among the simplest, followed in complexity by Context-Free Grammars (CFGs), Context-Sensitive Languages (CSL), and then Recursively Enumerable Languages. Each of these levels is distinguished by the types of production rules that are allowed in the grammars that generate the strings of the languages in question. There are schemes for parsing the strings of languages in each of the levels, which grow in complexity as the expressivity of the languages increase. The act of parsing is to recover a set of grammatical rules that are capable of generating the patterns found in some set of input strings. CFGs can be parsed by push-down automata (Autebert, Berstel, & Boasson, 1997), a type of abstract machine that is similar to an FSA, but has a stack that provides additional memory about the state of the processor. Recursively Enumerable Languages are parsed by Turing Machines (TMs) (Turing, 1936). The grammars of such languages can have extremely intricate patterns of rules; TMs are capable of computing any conventionally computable function (Herken, 1995)[35].

Accordingly, as the complexity of the grammars allowed by the classes of languages increase, the computational complexity of the parsing algorithms increases. This fact limits the ability of NLP technologies to employ completely general systems for parsing that exceed beyond a limited grammatical complexity. It has been shown that natural languages are at least as expressive as CFGs, with additional evidence that they incorporate CSL features as well. Since modeling CSL-level languages is largely

---

[35] There has been a great deal of work dedicated towards exploring the limits of Turing Machine "universality" and looking to extend the framework to encompass other types of computations: (Eberbach & Wegner, 2003). Also, see the concept of "hypercomputation" from biological (Calude & Paun, 2004) and non-biological (Ord, 2002) perspectives. It has been argued that human minds are hypercomputers (Bringsjord & Arkoudas, The modal argument for hypercomputing minds, 2004); however, this discussion of language assumes that a large part, if not all, of a program for human-level NLP can be implemented on a Von Neumann Machine-like computer (Bringsjord & Arkoudas, The modal argument for hypercomputing minds, 2004).

intractable, an intermediate-level called Mildly-Context Sensitive Languages (MCSL) have been proposed to approximate some of the more complicated patterns of natural languages. While further finer divisions of complexity in the Chomsky Hierarchy exist, the ones stated above are the most relevant for the study of natural languages. In terms of the overall goals of the NLP field, there is a drive to develop maximally expressive, yet highly constrained, systems that are capable of representing high degrees of complexity, while using assumptions to reduce the sizes of the involved problem spaces to computationally tractable dimensions.

10.2 Hidden Markov Models

For almost any phenomena that are interesting enough to model, the ability to develop a totally visible Markov Model is very rare. A full categorization of the mechanisms that contribute to the behavior of many complex processes, despite years of study, is almost impossible to obtain. The production rules behind the generation and interpretation of natural languages have been studied by thousands of linguists, for hundreds of languages, for hundreds of years. Yet, the detailed structure of the processes through which the human brain processes language is largely unknown. Many other complex systems, such as biological (Krough, Brown, Mian, Sjölander, & Haussler, 1994), industrial (de Almeida & Park, 2008), geological (Avesani, Azzoni, Bicego, & Orozco-Alzate, 2012), and meteorological processes (Ailliot, Thompson, & Thomson, 2009), have a "hidden" component to them. Alternatively, the behavior of processes might be influenced by more variables than is tractable to model. Perfect models of physical interactions in the Universe might require considering an intractable number of particles within the immediate vicinity of a target of interest; this has motivated the development of statistical models in physics that establish approximations for the cumulative and average behavior of large numbers of particles. In either of these cases of unavailable information, a distinction must be made between the roles of noise in observations of the phenomena and apparent randomness caused by gaps in theoretical understanding.

However, modeling these kinds of processes is possible because there may exist strongly supported theories of the mechanisms underpinning these processes that give

partial insight into their operation. Such knowledge can be leveraged to constrain the structure of a model through a set of intertwined assumptions that acts as a way to mitigate the effects of ignorance. The main draw of the HMM framework is its ability to provide this capacity with domain-independence: essentially, the same model can be used to study processes in every realm of science. Even though there have been many refinements and developments in the field of machine learning since the inception of HMMs, this framework remains highly ranked not only in frequency of use, but also in overall performance. A basic conceptual depiction of an HMM can be found in Figure 13.

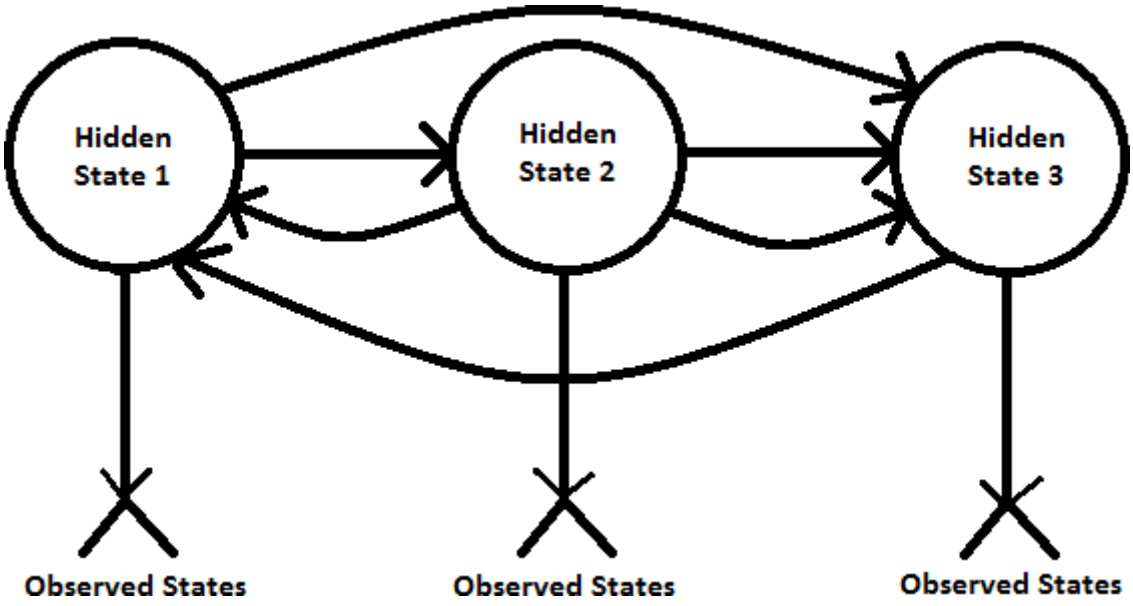

Figure 13. A graphical representation of a simple (unigram) HMM with 3 hidden states. In this model, all state transitions are possible. Note that hidden state to observed state links are directed. If this were an HMM employed for PoS tagging, the observed states would correspond to individual words in the corpus and the hidden states would correspond to the possible PoS tags.

Formal linguistics has been studied for millennia (Panini, 1987), but there are still no complete accounts of the mechanisms through which natural languages are understood and produced by humans. Even the fundamental physical processes of phoneme production by human speech organs and phoneme perception by human perceptual organs is a matter of ongoing research. However, it is valuable to produce models that reproduce phenomena while the theoretical underpinnings are still under development—

this process of theory refinement and hypothesis checking, according to the falsifiable view of science espoused by Karl Popper, is a never-ending series of refinements in any case. Computational models of linguistics serve the dual purpose of providing actual applied technologies, such as handwriting analysis or email filtering, as well as serving as both a springboard for and a constraint on the theories put forth by the various sub-disciplines of linguistics. Without the ability to produce models of a process that is imperfectly understood, these endeavors would be significantly curtailed. General findings and surveys of linguistic phenomena, however, are very useful in determining what specific processes are well-motivated enough to warrant computational study.

In an effort to decompose the gargantuan problems encountered in NLP, such processes are often approximated by discrete classes of phenomena are often considered in isolation and then either considered in a larger joint model or arrayed as components of a processing pipeline. Joint models are often problematic, since they typically exceed available computational resources to train in a tractable amount of time. Additionally, information about the interactions between the different processes in a joint model must be handled either by encoding them in the training data itself or by employing a modeling framework that can model these considerations implicitly. The first option requires a significant increase in the amount of training data in order to avoid overfitting; the second option requires larger models, which take more computational resources to train. Some approaches have been put forth to create large joint models, notably applications of the Markov Logic framework.

To circumvent the issues involved with producing joint models, a good deal of NLP research has invested in pipelined models, which have the benefit of neatly encapsulating aspects of language use in a plausibly separable way. This enables smaller models to obtain much more impressive results with less training data. On the other hand, interactions between different stages of pipeline models may be not be completely discounted. For instance, discourse-level processing might impact the parsing of individual sentences or phrases. Moreover, treating the different stages of an NLP pipeline as a one-way channel eliminates the ability to exert top down influences on lower-level processes and can mask the actual mechanisms of higher-level processes by making unrealistic assumptions, such as the lower-levels of processing will always

provide perfectly accurate, perfectly certain input. Top down and bottom up effects are known to work in tandem for many NLP tasks, so it is critical to incorporate multiple approaches to analysis (Tang, 2003). Despite these considerations, it suffices for the purposes of this study to focus in detail only on the simple, isolated-style of models that comprise most of what now counts as the NLP state-of-the-art.

Perhaps the most-studied example of a discrete NLP task that demonstrates how HMMs are typically employed to solve the problem of modeling the unobserved states of a complex process is Part of Speech (PoS) Tagging. The idea that for any given language there exists a limited and universal set of parts of speech, the common term for the morphosyntactic role that lexical items play in conveying meaning, has been accepted since ancient times, with proponents reaching as far back as speakers of classical Sanskrit (Matilal, 1990) and Greek (Law & Sluiter, 1995). Whereas these prototype theories are much simpler than their counterparts for most modern languages, the notion that parts of speech are as real a component of natural languages as words and sentences has persisted over time.

Although much development and thought has been invested into considering the rules by which the parts of speech are combined to produce meaningful expressions, no theory to date has encompassed a total description of the mechanics of natural languages—some researchers believe that such a program is antithetical to the nature of natural grammars themselves (Edlund, 1995). There have been efforts at producing simplified (Richards & Gibson, 1945) or artificial natural languages (Nicholas & Cowan, 2003) (Forster, 1982) to advance the study of formal linguistics, but, since natural languages are not closed and are in fact continuously evolving systems (Baugh & Cable, 2002), any attempt to define a static, inextensible characterization of a natural language is unrealistic. Even a sub-problem as limited as PoS Tagging is too broad to encode explicitly in a fixed set of constraints. Recall the description of the English Constraint Grammar in the knowledge-rich models section: thousands of hand crafted rules and exceptions are still not able to handle every possible case that is encountered in an English language corpus. Instead, fallback heuristics must be deployed in whatever cases still pose some amount of ambiguity to the tagger. Like all aspects of natural language, there are large amounts of exceptional and idiosyncratic cases that cause the explicit

enumeration of PoS tagging rules to be intractable. Furthermore, in some cases, relying solely on intra-sentence cues is not sufficient to disambiguate a particular PoS tag. A well-known example (Santorini, 1990) is: *The duchess was entertaining*. Here, *entertaining* can be cast as either an adjective or a verb participle, depending on the interpretation of the sentence. If one of those interpretations is actually supported by more evidence than the other, and there is no guaranteed that it could, then that information will be couched in the pragmatic, discourse-level of communication. For these reasons, Visible Hidden Markov Models are rarely, if ever, appropriate for realistic, sophisticated models of natural language phenomena.

Formally, the task of PoS tagging is to identify and assign a PoS role to every word in some natural language utterance. For a language L, there is a set of possible words, W, and possible PoS tags, T, which can be assigned to those words. Sentences that are members of L must be formed of words in W. A given sentence, S, contains n words that can be assigned an index, I, ranging from 1 to n. Each word in S, s_i, is assigned a PoS tag from T, which is denoted by t_i. Thus, a mapping of $W\,x\,T$ is established for each s_i. Although, by this definition, W would be limited to the actual vocabulary present in the training data, this condition is highly unrealistic for working with real natural languages. Not only would it be prohibitive to compile a data set with enough grammatical sentences[36] to cover all of the words in, say, English[37], it would also be pointless, as natural languages are continuously evolving new words and revising old ones (Aitchinson, 2001). Despite these factors, PoS tagging is usually considered as a

[36] Due to Zipf's Law (Zipf, 1950) regarding the distribution of frequencies for vocabulary words in natural language corpora, the expected number size of the corpus required to provide an exhaustive coverage of a given language's vocabulary is very large. A recent study by of the 'Google Corpus,' which consists of books that have been scanned into electronic form by the Google Books project counted approximately the same upper-estimate number of English words as produced by other methods: ~1,000,000 words. To reach this number, it was necessary to analyze the content of ~5,000,000 books (~4% of the total number ever published), a total of ~500,000,000,000 words, which illustrates what the dimensions for an exhaustive corpus for English might be (Michel, et al., 2011).

[37] The OED's conservative estimate of the current number of words in the English language is ~250,000 (Oxford Dictionaries, 2013). Upper estimates, for instance by the Global Language Monitor (GLM), reach ~1,000,000 words (Global Language Monitor, 2011). Although it is difficult to count the exact number of existing words in a natural language, thereby rendering meaningless any attempts to compare the sizes of natural languages, the OED suggests that English probably contains one of the largest vocabulary sizes of any widely spoken modern language (Oxford Dictionaries, 2013). However, agglutinative languages, such as Turkish, practically contain an infinitude of words, due to the large number of possible combinations of inflections and stems (Kurimo, et al., 2006). This issue has ramifications on the selection of NLP techniques used with agglutinative languages.

mostly trivial, knowledge-independent task. This approximation, as it does for most NLP tasks, produces compellingly good results for a small investment in modeling specificity. Almost every model used for this task considers this to be a sequential classification problem, which is an extremely convenient casting for machine learning algorithms. Many other NLP problems are also considered as classification problems for increasing the ease with which models for those problems can be created.

Like Visible Markov Models, HMMs are equivalent to a class of automata—in this case, Finite State Transducers (FSTs) FSTs are similar to FSAs, but operate over two alphabets, an input and an output, and thus can be used for labeling and classification tasks. More details about the expressive power and mathematical properties of FSTs can be found in (Mohri, 1997). Many NLP problems are naturally cast as classification problems, making models based on FSTs ideal for studying many types of linguistic phenomena with similarly structured models. PoS Tagging is inherently a classification problem, but it is also possible to cast other NLP problems as classification problems, as well. For instance, Named Entity Resolution (NES), which involves determining the mapping between mentions of an entity that occur in a discourse with some abstract entity that exists in some external concept space, can be considered to be a classification task that labels each mention to be a member of a class corresponding to the unique abstract entity to which it refers. Word Sense Disambiguation (WSD) is another problem that fits into the inference-as-classification paradigm.

10.2.1 Rabiner's HMM Tutorial

Perhaps the most widely cited and influential overview of the HMM framework, at least as applied to problems in the area of speech recognition is the treatment compiled in (Rabiner, 1989). Even though HMMs are often used to (and arguably better suited for) study continuous processes or processes in the physical sciences, they are just as frequently and, by some measures, successfully, purposed for discrete systems modeling, which is how text or token-based NLP would be classified. Many details about the motivation for using Markov Chains to model stochastic processes, how such models can be constructed and interpreted, and the algorithms for utilizing such models are provided in Rabiner's paper that are not within the scope of this document. However, due to the

popularity of the HMM framework and its many applications in the NLP field, a brief overview of the formal algorithms developed to interpret and train HMMs in practice will be developed below, according to the three problems for HMMs that Rabiner describes. In these sections, relevant criticisms and implications of using these algorithms will be discussed.

Rabiner represents a Markov Model as an expression of the model parameters. In keeping with mathematical notation for a set of parameters, Rabiner uses the symbol, λ, to refer to the entire model. Recall that there are three main parameters for a Markov Model: 1.) the matrix of hidden state transition probabilities, which Rabiner denotes as A, 2.) the matrix of hidden state to observed state emission probabilities, which Rabiner denotes as B, and 3.) the initial state of the model, which Rabiner denotes as π. Therefore, a generic Markov Model would be expressed as: $\lambda = [A, B, \pi]$.

10.2.1.1 Rabiner's Problem #1: Evaluating Likelihood of an Observation Sequence Given a Model

Although the evaluation problem itself is not frequently applied directly in NLP tasks, it is arguably the most basic of the ways to interact with an HMM and also provides insight into the other HMM algorithms. Formally, the evaluation problem is described as: determining the probability of a sequence of observations given some specified HMM. Even for small tasks, this problem would be unmanageable without the utilization of dynamic programming techniques that underlie what is known as the Forward Algorithm (Roger). The forward algorithm is similar in structure to each of the other standard HMM algorithms and is the conceptual precursor for the Forward-Backward Algorithm that is typically used to solve the evaluation problem in general. Dynamic programming techniques vast reduce the number of redundant computations required to use HMMs, which has contributed greatly to the widespread proliferation of this framework.

The problem statement, taken from Rabiner is as follows: given an observation sequence $\boldsymbol{O} = O_1, O_2, \ldots, O_t$ and a (Markov) model, $\lambda = [\mathrm{A}, \mathrm{B}, \pi]$, compute $\mathrm{P}(\mathrm{O}|\,\lambda)$, where t is the number of observed states in the sequence, A is the matrix of hidden state to hidden state transition probabilities, B is the matrix of hidden state to observed state probabilities, π is the matrix of initial hidden state probabilities, and $\mathrm{P}(\mathrm{O}|\,\lambda)$ is the

probability that the observation sequence was generated by the model. An example how this could be applied to NLP would be if an HMM was trained to recognize the letter patterns in a particular language, then that HMM could be used to evaluate how likely a given string of text belongs to that language. One could imagine that this would be useful for determining the most likely language origin of a particular text or perhaps the nearest extant language for a text in a dead language.

10.2.1.2 Rabiner's Problem #2: Decoding HMMs

After an HMM is trained, it can be used to perform inference. The most common inference algorithm for HMMs is known as the Viterbi algorithm (Viterbi, 1967), which also uses dynamic programing techniques to achieve computational tractability. The Viterbi algorithm is a type of Maximum A Posteriori (MAP) inference. MAP inference is the name assigned to a family of techniques that enable the estimation of a hidden distribution given an observable distribution.

The problem statement, from Rabiner is as follows: given the observation sequence $\boldsymbol{O} = O_1, O_2, \dots, O_t$ and a (Markov) model, λ, determine the hidden state sequence $\boldsymbol{Q} = q, q_2, \dots, q_t$ that is optimal to explain the observation sequence. There are multiple ways to define this optimality. The Viterbi algorithm does so by calculating the sequence of hidden state transitions that maximizes the overall probability of the sequence. If there are two hidden states, a and b, and three observed states x, y, and z, and the observed sequence is x→y→z, then there are 8 possible corresponding hidden state sequences[38]: a→a→a, a→a→b, a→b→a, a→b→b, b→a→a, b→a→b, b→b→a, b→b→b. To determine the probability of a sequence, the following procedure is used. For the first time index in the sequence, t = 1, the probability of being in each of the subsequence of length one is calculated for each of the hidden states, q_i: $P(subsequence_1) = \pi_i * B_i[O_1]$, where π_i is the initial probability of being in hidden state i and $B_i[O_1]$ is the emission probability from hidden state i to the observed at time 1. For all other time indices, the probability of being in that subsequence is computed by multiplying the probability of the subsequence that preceded the current time index by

[38] In general, there are n^t possible sequences, where n is the number of hidden states in the model and t is the number of observations. If some hidden state transitions are prohibited, this number will obviously be smaller by an according amount.

each of the next possible hidden states, q_i: $P(subsequence\ _j) = P(subsequence\ _{j-1}) * A_i[q_{j-1}] * B_i[O_j]$, where $A_i[q_{j-1}]$ is the transitional probability between the previous hidden state in the current subsequence and $B_i[O_j]$ is the emission probability from hidden state i to the observed at time j. When time t is reached, the subsequence with the highest probability is returned as the most likely sequence of state transitions to explain the observations according to the parameters of the current model.

In general, it is computationally expensive to have to calculate all of the possible sequence probabilities directly, so the Viterbi algorithm follows a greedy strategy of only exploring paths that could possibly lead to a better result than the current best path. This strategy, along with the dynamic programing element, which reduces the number of computations for requiring additional memory storage, is what allows HMM inference to be tractable.

Although the Viterbi algorithm is very powerful, its basic form lacks some desirable properties for certain tasks. For instance, a "beam-search"-style approach can to be used if the n-best results are desired instead of just the most probable one (Chow & Schwartz, 1989). PoS taggers could benefit from an n-best search specifically in the case when there is an ambiguous tag assigned to a word. This method would allow all of the possible alternative taggings to be returned, as long as they pass a given threshold. Yet, even the n-best variety of the Viterbi algorithm does not eliminate all of the limitations of this type of inference. The Viterbi algorithm is similar to other back-tracking search algorithms, such as DPLL (Davis, Logemann, & Loveland, 1962), which has been used to solve Constraint Satisfaction problems (Brailsford, Potts, & Smith, 1999). These types of searches suffer from an inability to take future inference into account to alter past calculations.

An example of why this capability is important is that it is impossible for the Viterbi algorithm to efficiently "penalize" paths that lack some expected global characteristic, such as "most sentences contain a verb." While the Viterbi algorithm could demote the overall probability of a sequence if it is discovered to lack a verb[39] after

[39] Some sentences, of course, may not have an explicit verb.

processing has concluded, this ability would not aid in the determination of how to correct the current inferred sequence so that it would contain a verb in the most likely place. Further changes to the sentence after the verb "replacement" would almost certainly have to be done. There is evidence that people have sophisticated, structured, heuristically approaches to successfully approximate such tasks (Hale, 2009)[40], but the Viterbi algorithm would have to use brute force to determine the best revised sequence.

This issue is similar in nature to the "intelligent" backtracking or backjumping problem that is encountered with other types of search algorithms (Prosser, 1993). Moreover, it has been demonstrated that humans appear to be capable of this kind of post facto revision in a highly efficient manner. An additional shortcoming of the Viterbi algorithm is that it meant to be implemented to run after a discrete learning phase has been completed and that it needs to wait for the explicit processing of all competing paths to make final commitments. It has been shown that humans use their predictive capability to guide learning (Tobler, 2006). Humans also may "jump to conclusions" in an attempt to simplify complex problems or to use simple patterns that have proven effective in previously encountered situations (Kahneman, 2011). Despite the fact that cognitive biases may result in decreased accuracy, the human mind is predicated on satisfaction rather than optimization (Simon, 1956)—it could be the case that the imperfection of human cognition are in place to allow difficult problems, such as NLP, to be solved in real-time.

10.2.1.3 Alternative Solutions to Rabiner's Problem #2: Beyond the Viterbi Algorithm

There is an alternate interpretation of the Maximum a Posteriori (MAP) inference problem posed when decoding an HMM. Instead of maximizing the probability over each entire sequence of model state transitions, the probability of each individual model state can be maximized. To understand how these methods might produce different results, consider the following modeling situation. There is a Markov Model, M, with three possible states, let these be a, b, and c. There are five distinct observable states, call them v, w, x, y, z. There is a sequence of observed variables: v, v, w, z, y, x. Assuming

[40] Notably in the recasting of garden path sentences.

that M has been trained and thus has set values for its parameters, a MAP inference algorithm based on the Viterbi algorithm would maximize the probability over each of the possible combinations of hidden state transitions: $a \to a \to a \to a \to a, \ldots, c \to c \to c \to c \to c$.

The properties of the dynamic programing used by the Viterbi algorithm of course greatly reduce the number of computations required to determine this maximum probability. Say that over all, the transition sequence: $a \to b \to c \to b \to a$ has the highest probability. It may be possible, however, that there are subsets of this transition that have a lower probability than an alternative transition of states. For instance, in the sequence selected by the Viterbi algorithm, the probability of the first hidden state[41] being "a" might not be the locally optimal choice. This situation can occur if the probability of paths starting with a different state are higher at that point, but all subsequent transitions from that state are very low. For instance, the probability that "b" might be first hidden state could be 0.5, whereas the probability of a could be 0.4. If the transitions $b \to a$, $b \to b$, $b \to c$ all have significantly lower probabilities that $a \to a$, $a \to b$, $a \to c$, then it is conceivable for the Viterbi algorithm to select a path that begins with a. However, it is acceptable to define the MAP result as maximized over each individual state rather than the entire sequence of states, in which case, b would be selected over a. This formulation is usually called "posterior decoding" (Batzoglou, 2007).

For some applications, posterior decoding may be even more accurate than the Viterbi algorithm because it helps to select between transition sequences that are very close in probability (Fariselli, Martelli, & Casadio, 2005). The one drawback of posterior decoding is that it does not necessarily obey the underlying grammar that is implicitly represented by the HMM (i.e. it can allow zero probability transitions). Due to this issue, posterior decoding is not typically used with NLP tasks. Additionally, there are other formulations of how to perform inference with HMMs, some of which are explained in (Fariselli, Martelli, & Casadio, 2005).

---

[41] This is computed as a function of the initial distribution of model states and the emission probability from the current state to the first observed state.

10.2.1.4 Rabiner's Problem #3: Training HMMs

One of the purported benefits of the HMM framework is that there are efficient algorithms for estimating the parameters of its models. At the onset, the problem of training an HMM to reflect the properties of a data set is even less tractable than the evaluation problem identified above. Although the number of parameters for a simple HMM naturally scales up extremely fast with the size of the model, a technique known as Expectation-Maximization (EM), of which the Baum-Welch algorithm[42] (Welch, 2003) is the standard implementation. While this training algorithm is capable of fitting the parameters of HMMs to very large data sets, it is also known to lead to parameter choices that satisfy local, not global optima (Papanicolaou, 2011). Additionally, the efficiency of the Baum-Welch procedure does not allow HMMs to contain an extremely large number of states. For many of the refinements of HMMs that significantly increase the state-space of models.

The problem statement, from Rabiner is as follows: given a (Markov) model, $\lambda = [\mathrm{A}, \mathrm{B}, \pi]$, adjust the model parameters to maximize $\mathrm{P}(\mathrm{O}|\,\lambda)$, where A is the matrix of hidden state to hidden state transition probabilities, B is the matrix of hidden state to observed state probabilities, π is the matrix of initial hidden state probabilities, and $\mathrm{P}(\mathrm{O}|\,\lambda)$ is the probability that the observation sequence was generated by the model.. This is the learning problem for HMMs and as such, it is the most difficult to solve computationally and the most important in terms of model creation. To train a HMM to learn how to tag words with their PoS, the typical practice is to run an EM algorithm to set the transitional and emission probabilities. For a unigram HMM, the hidden states are typically defined so that there is one hidden state per part of speech. A segment of a tagged training corpus would appear as: $(This : DET)$ $(is : V)$ $(sentence : N)$ $(one : N)$ $(.:.)$ $(Sentence : DET)$ $(two : V)$ $(follows : N)$ $(sentence : one)$ $(.:.)$.

However, it is also possible to perform semi-supervised or unsupervised learning using the EM algorithm. A basic introduction to unsupervised learning for HMMs is provided in (Statos, 2012). The accuracy obtained by unsupervised method is, unsurprisingly, less than that of supervised methods. For example, a state-of-the-art

[42] Sometimes called the Forward-Backward algorithm.

HMM PoS tagger that trains using an unsupervised EM algorithm (Goldberg, Adler, & Elhadad, 2008) obtains only 91.4% accuracy. This may not seem like a undesirable trade-off between the moderately higher accuracies achievable by supervised HMMs and needing a hand-annotated corpus, but since the field is extremely competitive in terms of small metrical improvements and because this is a per-token, not a per-sentence measure, the difference is enough to discourage the use of unsupervised learning methods in this arena. Furthermore, the unsupervised HMMs was supplemented by providing the model with serendipitous initial conditions, as well as hand-coded changes to the model, both of which defeat the purpose of using knowledge-lean methods over knowledge-rich ones.

10.2.1.5 Alternative Solutions to Rabiner's Problem #3: Beyond the ML Training Algorithm for HMMs

Alternatives to the EM-based Baum-Welsh algorithm that avoid some of its undesirable properties have been developed for training HMMs. For instance, Maximum Mutual Information HMMs present a discriminative learning method that attempts to avoid local minima by recasting the training of HMMs as a reflection of the information associated between different events in the training corpus (Chow Y.-L. , 1990). In addition to the possibility of learning a better model, Maximum Mutual Information algorithms allow certain types of complex data to be learned that are not handled well by the EM algorithm. These data appear to be mostly important for speech processing, but it is possible that there are analogs for other NLP tasks as well.

On the other hand, Bayesian methods have been proposed for learning in HMMs with some success. However, there are still certain circumstances in which the EM algorithm is capable of competitive performance (Gao & Johnson, 2008). Bayesian methods appear to be geared towards smaller model sizes (Ravi & Knight, 2009).

Additionally, schemes to use genetic algorithms to select differently trained HMMs in an attempt to find a version of the model that has not fallen into a state of local extrema have been developed (Xiao, Zou, & Li, 2007). This method demonstrates that the Baum-Welch method is not guaranteed to optimize the training of HMMs. Applied to the task of information extraction, the HMM supplemented by genetic algorithms attained an accuracy of 84.5%, whereas the standard training method produced a 71.0% accuracy.

Despite the known shortcomings of the EM algorithm, none of these alternatives provides a truly satisfactory resolution of the issues. It is a common theme in NLP, especially in the knowledge-lean paradigm, to attempt a large number of approaches in an attempt to find the optimal variant for a particular problem situation. This can be seen as an oblique way to "sneak" knowledge-engineering-like techniques into the knowledge-lean paradigm. Claiming that knowledge-lean methods are more flexible, general, and more easily modified than knowledge-rich techniques is misleading, because this ignores the need to develop adaptive programs *for each individual case in the problem spaces*. Such practices are apt to lead to misunderstandings about the relative fitness of technologies and will likely serve to retard the progress of the NLP field towards the goal of human-level performance.

10.2.2 Applications to NLP

HMMs were one of the first statistical modeling frameworks utilized to quantitatively study natural language phenomena. Originally, HMMs proved to be useful tools for transcribing human speech to text. The acoustical physics behind the generation of phonemes was able to be captured well enough by HMMs to enable the construction of the first speech understanding systems, even under the restrictive nature of the Markov Assumption. This success paved the way for applying HMMs to NLP problems that involved the parsing and understanding of text. One of the most basic NLP tasks is Part-of-Speech (PoS) tagging, which aims to label each word in an utterance with the morphosyntactic role that that word plays in contributing to the semantic interpretation of the utterance. Even though it is possible for long-distance dependencies to influence the PoS tags in English sentences, the strength of English's rules of word order and phrase construction permit HMM-based methods to exhibit great success at solving this problem. Because many NLP systems rely on PoS Tagging as an underlying subroutine, this task has actual practical significance. It is also useful to create HMM models that can perform PoS tagging in other natural languages, but some languages pose difficulties for the basic framework. A depiction of an HMM trained on data for a basic English PoS tagging task can be found in Figure 14.

To understand more completely how an HMM can be leveraged to perform PoS tagging it is useful to understand how elements of the PoS tagging task map onto HMM parameters. Consider that Hidden Markov Models have the following general parameters: a set of observed states, a set of hidden states, a set of initial state probabilities, a set of transition probabilities between hidden states, a set of emission probabilities between hidden and observed states. These parameters map onto parts of the PoS tagging problem in the following way. The observed states correspond to words in the target language. The hidden states correspond to part of speech tags in the target language, the initial state probabilities correspond to the most likely tag for the first word in the average sentence. The transition probabilities correspond to constraints that encode the chance that a particular tag follows another tag in sequence (e.g. DET at position i-1 $\Rightarrow$(.67) NN at position i). The emission probabilities correspond to constraints that encode the chance that a particular tag is tagging a particular word (e.g. DET at position i $\Rightarrow$(0.33) *The* at position i).

Attempts have also been made to perform syntactic parsing with HMMs, even though the state-of-the-art approaches use a fundamentally more complex class of model known as a Probabilistic Context Free Grammar (PCFG), which will be discussed in detail in a later section. The task of syntactic parsing a natural language is an extremely difficult problem, even if training data are perfectly annotated, the text is entirely error-free, and ungrammatical uses are minimized. Despite the inherent "regular-expression" restrictions on HMMs, it is possible to use them to parse sentences (Hogenhout & Matsumoto, 1998). Additionally, HMMs can be used for semantic modeling, as they have been deployed to solve word-sense disambiguation problems (Molina, Pia, & Segarra, 2002).

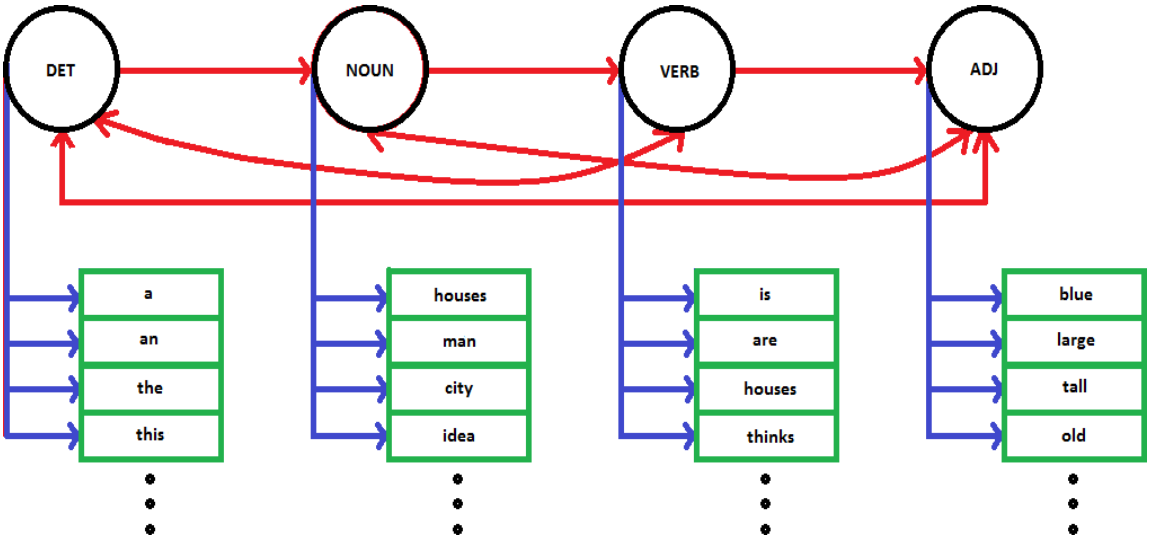

Figure 14. An HMM model trained on a PoS tagging task.

10.2.3 Further Development of HMMs for Acoustic Language Processing and Non-NLP Applications

Although an in-depth discussion regarding the application of HMMs to analyzing acoustic representations of natural language is beyond the scope of this study, it is worthwhile to mention some of the background of the development of HMMs for this purpose, as many of the innovations that have been developed for speech-processing could, theoretically, also be applied to text-processing. Since some of the refinements of HMMs to enable speech-processing have not or have only in a limited way been applied to other NLP applications, one may object that, perhaps, the criticisms levied against the statistical approach may be invalidated. However, the properties of the underpinning mathematics of HMMs that make them poorly suited for studying a complex, highly-structured and context-dependent phenomenon as natural language remain unchanged due to these refinements.

In the spirit of completeness, the following references are provided for a sample of such papers: (Kelly, Dillingham, Hudson, & Wiesner, 2012) (O'Neill, Barlow, Safranek, & Beige, 2012) (Fujimaki & Hayashi, 2012) (Johnson & Willsky, 2013) (Zhang, Qui, Zheng, & Yang 2012). All of these papers were published no earlier than 2012. As of 4 FEB 2013, searching for papers containing the text string "Markov Model" that were published since the beginning of 2013 results in 1,030 hits. Clearly, it

is intractable to thoroughly review all of these papers, even if a small percentage of them were actually relevant to the application of Markov Models to NLP problems. While there is nothing inherently wrong with exploring variations on a technological theme, it is distracting if that particular type of technology cannot perform the tasks required of it. By analogy, atmospheric balloons can reach great heights in the Earth's atmosphere, but no variation of balloon technology will ever all a balloon to reach the moon unaided.

10.3 Performance of Markov Models

Like most NLP frameworks, HMMs are evaluated based on a metric that uses a definition of accuracy that is based on a ratio of measures of precision and recall. These measurements are motivated by the notions of Type I (false positives) and Type II errors (false negatives) that are used in evaluating the success of hypothesis testing techniques for classification tasks (Powers D. M., 2008). In this context, precision is defined as the ratio of true positives to the sum of the true positives and the false positives: $P = \frac{True\_Positives}{True\_Positives + False\_Positives}$. Recall is defined as the ratio of true positives to the sum of true positives and false negatives: $R = \frac{True\_Positives}{True\_Positives + False\_Negatives}$. The F-score, which combines precision and recall, is often used as the overall gauge of accuracy for applications of information processing (Makhoul, Kubala, Schwartz, & Weischedel, 1999): $F1 = \frac{2*P*R}{(P+R)}$. This quantity indicates how well a given model avoids Type I and Type II errors. In some papers, both labeled and unlabeled precision and recall scores are reported. All of the criticisms that were raised against this type of local-structure-based statistical measures in the section on NLP statistics earlier in this document hold for HMMs. Because HMMs are inherently based on localized pattern learning, it is no surprise that these metrics are used for the evaluation of such models. However, since this type of measurement is the standard of the field, success or failure on a task is more or less purely determined by this number (there is sometimes a concern over the computational efficiency of the methods used (e.g. see Section 4.2 in (Bikel, Miller, Schwartz, & Weischedel, 1997)).

By examining the reported F-scores of HMMs used for various NLP tasks, it becomes clear why this approach is popular among statistical modelers. Taken from the

data compiled from the ACLWiki's state-of-the-art pages (Assoc. of Computational Linguistics, 2013), the following table illustrates how well HMMs perform on a variety of NLP subtasks (see Table 3 below).

Table 3. HMM approaches applied to various NLP problems and the maximum reported accuracies for those applications.

NLP Problem	Accuracy (F1-Score)	Citation
PoS Tagging	96.46%	(Brants, 2000)
NP Chunking	95.23%	(Shen & Sarkar, 2005)
Information Extraction	92.90%	(Seymore, McCallum, & Rosenfeld, 1999)
Japanese Word Segmentation	91%	(Papageorgiou, 1994)
Word Sense Disambiguation	60.20%[43]	(Molina, Pia, & Segarra, 2002)
Phrase Extraction	92%[44]	(Feng & Croft, 2001)

10.4 Favorable Properties of Markov Models

Aside from its effort to increase understanding of how statistical modeling approaches can be applied to isolated aspects of cognitive science and linguistics, the detail of the above discussion has been included in this report to illustrate why this paradigm is so prevalent in the NLP field. Many of the reasons why Markov Models are considered to be an appealing bellwether for NLP technology can be applied to other statistical formalisms, as well. Some analysis of these apparently beneficial properties is required in order to understand how the statistical approach has been so successful at out-seating knowledge-rich approaches to cognitive modeling. One of the primary benefits

[43] According to the source, this is a competitive result in comparison to other state-of-the-art models.
[44] Computed from the cited recall of 91% and precision of 93%.

of most statistical models, especially true of Markov Model-based approaches, is their conceptual simplicity. The mathematics and algorithms behind these techniques are also very efficient, at least compared to naïve approaches. This efficiency, in turn, is domain-general and can be applied to data sets with any type of content, provided that the data matches the assumptions about the characteristics of the probability distribution incorporated in the model.

As seen above in the sections on HMM decoding, there is also some degree of flexibility in defining, for an individual model, what criteria are used to assess the optimality of a state sequence. Because of the simplicity of the structure of HMMs, there is no need for modelers to be an expert in linguistics or in the particular domain for which they are developing NLP applications (i.e. biomedical documents). Some knowledge-lean approaches require some amount of expert knowledge to create suitable initial models. For instance, the creation of Bayesian Networks relies on model design input from experts, which often leads to model development taking the form of a cycle of trial-and-error driven revisions (Wiegerinck, 2005). While there are automatic methods for learning such model structures automatically, it turns out that the individual methods devised for doing so are tailored to specific task demands (Daly, Shen, & Aitken, 2001). Therefore, some kind of expert human-based knowledge is required in somewhat for more structured types of models that is unnecessary in basic HMM modeling.

Additionally, the fact that HMMs can work in unsupervised, semi-supervised, or supervised learning environments with simply structure annotations is a great advantage for researchers. Developing complex annotations for corpora is a difficult and expensive task, which often requires expert-knowledge in itself. By simplifying and reducing annotations, or even eliminating them altogether in the case of unsupervised learning, the cost of the development and deployment of HMMs can be minimized. Therefore, it is fairly straightforward and highly feasible to produce a large amount of cutting-edge research by using this one formalism.

Finally, it is worth noting that HMMs are accurate enough to warrant their use in NLP systems and have stood the test of time as a perennially employed algorithm for many types of problems. It is unlikely that a suitable replacement for HMMs (from the perspective of certain problems that HMMs are not optimal to solve) will be developed in

the near future. Because of the tried-and-true nature of HMMs, they are worthwhile for many researchers to learn both because that knowledge will aid their understanding of a large amount of other work in the field and also because using HMMs in their own work will ensure that reviewers, as well as potential citation-makers, will have an easier time accepting their research.

10.5 Metrics Used in HMM Applications

The metrics developed to assess the performance of HMM applications are similar to those used with many other types of NLP models. Accuracy as defined as a ratio of precision and recall is the standard statistic reported by NLP researchers and is used in essentially all comparative surveys of the different approaches. Although other considerations, such as efficiency or more desirable mathematical properties, drive the continued development of new statistical NLP techniques, the desire to produce models that perform with slightly higher accuracy than the previous state-of-the-art has become a principal concern. This trend is understandable from the perspective that the accuracy measure is the most salient result from an NLP "experiment" and appears to be a meaningful assessment of model performance. Additionally, it is possible to compare the accuracies of any NLP system with other NLP systems, including humans. Systems that perform syntactic parsing of natural languages are graded on their differential performance in a number of contests (Doddington, et al., 2004) (Grishman & Sundheim, 1996) (Voorhees & Harman, 1998).

10.6 Examining Corpus-Based Approaches to NLP

The success of HMMs in the NLP field led to a great increase in the number of corpus-based, knowledge-lean approaches used to study natural languages. As will be discussed below, there are a large number of corpus-based approaches besides HMMs, each of which is subject to the following set of criticisms. Essentially, the whole of the state-of-the-art is now involved with corpus-based statistical approaches, which is why the appropriate perspective needs to be taken regarding the overall potential for human-level natural language processing to be derived from corpus-based methods. On one hand, there are clear benefits that suggest that corpus-based NLP deserves its status at the

forefront of the field. For instance, there are a large number of suitable training corpora for a large number of the more common natural languages. Although many of the best developed corpora are for English, there is an increasing supply of corpora for other languages (Lee, 2010). This document is written from the perspective of a native English speaker, and so will focus almost exclusively on English-based corpora.

Among the most widely used English-language corpora are the Penn Treebank (Marcus, Marcinkiewicz, & Santorini, 1993), especially its Wall Street Journal (WSJ) subsection (Paul & Baker, 1992), which is composed of articles from the Wall Street Journal that are written about a number of different subjects, and the British National Corpus (Aston & Burnard, 1998), which consists of written text and transcriptions of spoken words. Corpora based on Twitter (e.g. The Edinburgh Twitter Corpus (Petrovic, Osborne, & Lavrenko, 2010), the Google web text n-gram corpus (Brants & Franz, 2009), and the Google Books corpus (Davies, 2011) are extremely large scale and can be used for purposes beyond those traditionally associated with Treebanks (e.g. PoS Tagging and parsing). Microsoft has also released a web-based n-gram corpus (Wang, Thrasher, Viegas, Li, & Hsu, 2010). The large selection of corpora that have been cultivated from online repositories new reinforces the notion that Internet-based resources are capable of providing the next generation of NLP corpora.

Twitter-based corpora can even be used to mine semantic and pragmatic information from text, as was demonstrated in a study that attempted to induce the time scales of activities described by verbs (Williams & Katz, 2012). In doing so, this study determined that the average time frame of the activities denoted by words could be learned via machine-leaning techniques, thereby opening up the possibility for learning other semantic features via the corpus-based methodology. For instance, it was possible to learn a distinction was between instantaneous and habitual descriptions of activity, which might have differing implications for the expected time frame of the types of behavior described by the verb. This type of information is useful in establishing operational definitions for various word senses. For instance, the verb "search," when used in the episodic sense, had a typical time-scale of minutes to hours; in the habitual sense, the average time-scale for "search" was on the order of years. This kind of information could aid in establishing clues for determining word senses from context.

The vast amount of easily accessible text from Twitter also enables researchers to make much more refined models of the time-scales associated with specific verbs (Williams, 2012). A similar application of the Twitter corpora has been to utilize the dialectical nature of twitter posts and reactions to Twitter posts to mine event and entity descriptions (Popescu, Pennacchiotti, & Paranjpe, 2011), as well as to perform sentiment analysis and opinion mining (Pak & Paroubek, 2010).

Google has released several blog posts about the various applications for their recently released large-scale corpora. The Google Books corpus has now also been annotated with PoS tags and syntactic phrase structure information (Lin et al., 2012). Because of the massive extent of the Google Books corpus—it purportedly contains 6% of all published material in English from the start of the 19th century onwards—it is certainly a valuable tool for NLP projects. Moreover, Google, perhaps more than any other company, has realized the huge wealth of information that is available from documents on the internet. At least portions of both of the Google corpora have been made available for researchers. The Microsoft web n-gram corpus is similar to the Google web n-gram corpus, except that it provides additional information about the source documents of the corpus.

Despite the versatility and economy of the corpus-based approach to NLP, there are a number of serious drawbacks associated with it. First, it is difficult to establish any kind of systematic training regimen from the text. It has been shown that both humans (Markman, 1994) and machines (Elman, 1993) learn best when presented with structures that increase in complexity—this allows them to learn examples that increase in difficulty. Without direct human evaluation of corpora material (likely intractable for a corpus the size of 6% of all English language publications), it would be impossible to arrange for such a sequence of trials to determine if that type of learning technique is required for robust NLP acquisition. Additionally, it might be useful to separate out the styles of text (e.g. in terms of format, formality, genre, etc.) to train models to learn how different types of language use and the associated contexts impact the understanding of language.

Another major deficiency with corpus-based approaches is that, even when they are annotated, those annotations are necessarily knowledge-lean. It is difficult or

impossible to learn complex structures from simple structures by applying leaning methods that are not self-modifiable. Finally, corpora-creators are inevitably incapable of foreseeing the future of NLP processing technology or even the full range of possible uses for their corpora. This inability to evolve with demands of NLP developers is discussed in: (Burnard, 2002). When the resources and tools available to a scientific community cannot mature naturally with the state-of-the-art progress in that field, the rate of development will certainly be significantly slowed.

10.7 Smoothing

One major problem for HMMs is that they have no information about patterns that they encounter during evaluation that were not present in the training data. For the PoS tagging problem, this could arise in the form of a word that appears in an evaluation corpus that was not included in any of the training data. Techniques known as "smoothing" have been developed that can account for these gaps in the HMM's representation of the modeled distribution (Lember, 2011). However, it should be noted that the problem of smoothing is far from being conclusively solved. There is much debate over how to motivate the choice of one smoothing technique over another.

It is also difficult for smoothing to account for "corrigible" vs. "incorrigible" mistakes in natural languages. Abnormal and deliberately ungrammatical constructions are used routinely by natural language speakers to convey a particular nuance of emotional or semantic meaning, as is commonly found in rhetorical styles of utterances. The famous quip by professional boxing manager Joe Jacobs, "we was robbed," has been the subject of scrutiny by observers in the field of linguistics (Quinn, 1993). By examining only purely grammatical English language texts HMMs learn a very stilted portion of the total range of the language. This is compounded by the fact that even in assiduously edited texts, there are chances for random errors; the error rate increases as the size of the text increases, which is a problem, since, in practice, large corpora are required to train statistical models of natural languages. To compound this difficulty further, it is possible for errors to look locally fine, but only on examination of an entire clause, sentence, or passage would be revealed to be inconsistent. In many cases, individual natural languages lack a sufficiently fine-grained vocabulary or the necessary

flexibility in construction formulation to express certain types of knowledge about the world. None of these issues can be solved with presently available smoothing techniques.

10.8 Relation to Dynamic Bayesian Networks

Dynamic Bayesian Networks (Murphy K. P., 2002) are a form of graphical inference models that use Bayesian inference (O'Hagan, Forster, & Kendall, 2004) and learning (Heckerman, Geiger, & Chickering, 1995) techniques to fit patterns in data. Like HMMs, DBNs can be conceptualized as Directed Acyclic Graphs (DAGs), in which the nodes represent random variables that contribute to the behavior of a stochastic process and the direction of links determines the direction of the causal relationship between dependent variables. DBNs are also capable of directly modeling sequential data, because their state at previous time points conditions their state at later time points. However, DBNs are subject neither to the Markov Assumption nor to the restriction that observable variables cannot condition other observable variables. This enables a much wider range of structures that can be modeled by DBNs in comparison with HMMs. See Figure 15 for a depiction of a generic DBN.

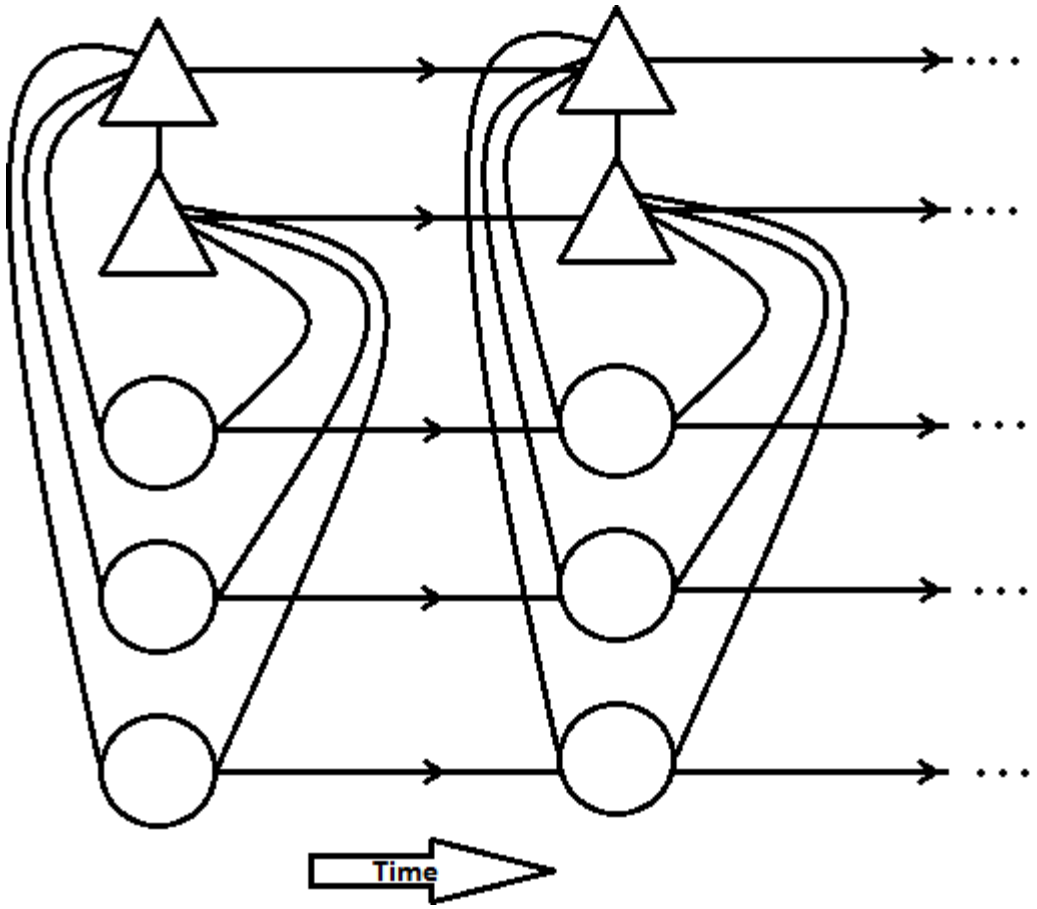

Figure 15. A generic DBN

DBNs have also been used to capitalize on a property of natural languages known as "tree locality (Savova & Peshkin)." One of the difficulties in using regular-expression-equivalent formalisms is that they are inherently limited in the amount of context and long-distance relationships that they can encode. However, it is known that a large number of features of natural languages are non-local in that they contain relationships between lexical information in an utterance that are non-adjacent. As an example, consider the case of a modifying phrase that separates a noun from its associated verb: *The man went to the store, The man that I saw went to the store, The man that I saw yesterday went to the store,…, The man that I saw yesterday with the black hat in the city park who was walking his dog and talking on his cell phone to his daughter's friend from Boston, Massachusetts went to the store.* A normal FSA-equivalent modeling method would not be able to accommodate the fact that arbitrarily

long phrases can be placed within particular grammatical constructions in a natural language[45].

10.9 Overview of Drawbacks of Markov Models

HMMs have undoubted proven to be a useful formalism for statistical modeling, provided that they are employed for appropriate tasks in appropriate ways. The results mentioned have shown that it is possible to approximate human level performance for PoS Tagging in English using HMMs, but only when somewhat misleading metrics are used for the assessment. While it may seem unlikely that the discrepancy of 4-5 percent of perfect accuracy in PoS tagging would prohibit the development of sufficiently capable NLP technology, the evidence is in the fact that the best available systems are either very rigid or restricted to extremely narrow domains. Such applications are difficult to extend naturally and may require vast amounts of training on precisely structured data to improve their performance. This inflexibility stems largely from the inability to rely on pipeline-based models of NLP in practice, since errors propagate through the stages of the pipeline. Without nearly flawless PoS tagging, it is unrealistic to expect that a simple parsing algorithm that requires error-free input from lower levels of processing to produce accurate recovery of the deep syntactic structure of the input utterances.

Since NLP systems derive much of their semantic analysis from lower-level syntactic parses, it is infeasible for human-level performance to be achieved without a solid foundation. Many of the problems of constructing syntactic structures for NLP utterances arise out of the presence of ambiguities that cannot be resolved without some degree of the context of the utterance, which is a massive space of information that includes a potentially vast amount of world-knowledge and may rely on diverse mechanisms for commonsense reasoning. Even some of the most rudimentary semantic processes, such as identifying predicate arguments, are highly reliant on accurate

[45] Note: some languages actually do not appear to exhibit this property structural complexity (e.g. Pirahã (Everett, 2005)). Nevertheless, these languages contain referential words and constructions that require some handling of world knowledge to comprehend. The reliance on complex world knowledge structures for parsing is beyond the scope of most, if not all, knowledge-lean approaches to NLP. This includes DBNs.

syntactic parses (Gildea & Palmer, 2002). Moreover, there are a number of languages that do not share the same types of distribution of information that are prevalent in English (Arslan & Patan, 2009), which leads to difficulties in applying HMMs as a universal tool for low-level language acquisition. It is also known that human cognition relies heavily on both top-down and bottom up influences (Solso, MacLin, & MacLin, 1995). A standard HMM has no way of incorporating the results of higher-level processing.

Although the simplicity of the Markov Modeling framework leads to its numerous attractive characteristics, many of these benefits have a corresponding downside. For example, the production of massive corpora is possible, but the information obtained from such sources provides only very general patterns that are reflections of much more detailed The contribution of these more detailed factors to the behavior of a process is difficult to capture directly in a precisely-engineered system, which is why many of the rule-based approaches to NLP have proven unsatisfactory. To avoid having to bring this level of unavailable expert knowledge to bear during model construction, it is necessary to examine a large amount of very broad trends that implicitly capture the average, cumulative results of the influences of those factors. Markov Models are extremely limited in the depth of knowledge that they can incorporate and therefore are prone to make errors whenever there are locally ambiguous constructions. In natural languages, there can be arbitrarily long-range dependencies and nested clauses, both of which are beyond the capability of HMMs to model. The simplicity of HMMs allows them to admit tractable learning and inference procedures, yet, this same property prevents them from scaling to human-level NLP tasks. At this point, the only way to improve HMM results is to defeat the underlying limitations of the model by going beyond the HMM framework. This process can involve operating in knowledge-lean arenas or can involve hybridization with expert-systems-like techniques to patch known problems in HMM models with hand-engineered augmentations.

As will be revealed below, it is possible to use a discriminative model in a much more targeted way by making use of complex features that are posited explicitly to allow a much more transparent model. Although it is possible to improve the results of HMMs through the use of discriminative techniques, the main drawback of discriminative

models is that they are constrained to use supervised learning algorithms. Because there is a considerable debate as to the level of supervision that humans received while learning natural language grammars, discriminative modeling techniques may have limited cognitive plausibility.

10.10 Extensions to HMMs

In order to address some of the inherent shortcomings of the basic HMM framework, a number of extensions along various dimensions have been proposed. Most of these assumptions have been: 1.) aimed at relaxing the Markov Assumption, so that more than just the previous hidden state can condition transitions from the current hidden state, or 2.) aimed at increasing the complexity of hidden states to expand the expressivity of HMMs beyond the realm of Finite State Automata. The most common types of extensions, and the ones that will be covered here, are: 1.) increases to the model memory, 2.) manipulations of the underlying model mathematics, 2.) increases to model structure complexity, and 4.) increases to the complexity of modeled information.

While each of these extensions does boost the power and, for some applications, the accuracy of HMMs, they are not panaceas. First, even with the most advanced extensions, HMMs are still not expressive enough to capture all of the nuances of natural languages. The HMM framework is simply not sufficiently nuanced to accommodate the knowledge-rich data representations that are necessary for human-level NLP. Second, as extensions to HMMs grow in complexity, they begin to encounter new problems from which standard HMMs are spared. HMMs are based on the idea that very simple models are capable of capturing a surprisingly large portion of the causal structure of many kinds of stochastic processes. But, this simplicity is a double-edged sword. On one hand, these simplifications, such as the Markov Assumption, are what make HMMs computationally tractable. On the other hand, these simplifications severely limit the expressivity of HMMs. Heretofore, there have not been any extensions to the HMM framework that preserve tractability while also significantly increasing model expressivity; modifying the original assumptions of HMMs invariably leads to a model state space explosion, which ultimately has led researchers to explore other kinds of stochastic models.

10.10.1 Increasing Model Memory

Perhaps the most common extension to the basis HMM formalism is the augmentation of the Markov Assumption to allow longer sequences of data to be considered during the conditioning of the model. Although this is at odds with the idea that an HMM is memory-less except for the previous state transition, this stipulation can be applied to longer sequence of states in the following way. Consider an HMM that is designed to perform PoS tagging. A basic HMM is of order-1: it learns from unigrams (i.e. word sequences of length 1) and individual PoS tags. However, it turns out that local dependencies beyond the current word (or other natural language structure) are very influential in determining how meaning is constructed. Some knowledge of this principle can be obtained by reading about the linguistic phenomena of constructions (Fillmore, Kay, & C., 1988) and phrases (Katz S. M., 1996). More technical treatments of locality[46] in language can be found in treatises about syntagmatic and paradigmatic analyses of multiword structures (Jacquemin, 1999).

Because higher-level information structures in natural languages can often be decomposed into components that are distributed among lower-level structures and because these lower-level structures might not be restricted to one structural unit (e.g. word), it is necessary to expand the set of possible influences on the generation of each word in a sentence. Although bigram and trigram HMMs are capable of exceeding the performance of unigram HMMs, they are still not capable of representing arbitrarily long-distance dependencies that are a major factor in obtaining human-level NLP performance. Therefore, even though HMM-based approaches might score highly in an absolute sense of evaluation, it is the remaining error that they cannot correct that is actually the most important to comprehending the larger-scale mechanics of natural language grammar. As such it is not only the case that HMM-based approaches will soon reach an (albeit high) performance ceiling that they will not be able to overcome without knowledge-engineering-like solutions, but also it is likely that this remaining gap in processing is correlated with the features of natural language that underlie its special

[46] Note local doesn't necessarily imply temporally local—languages that have loose word order can be considered to exhibit locality in an abstract sense (c.f. the notion of "tree locality" in parsing (Chiang & Scheffler, 2008)).

usefulness as an efficient mode of communication. A basic conceptual depiction of a higher order HMM can be found in Figure 16 below.

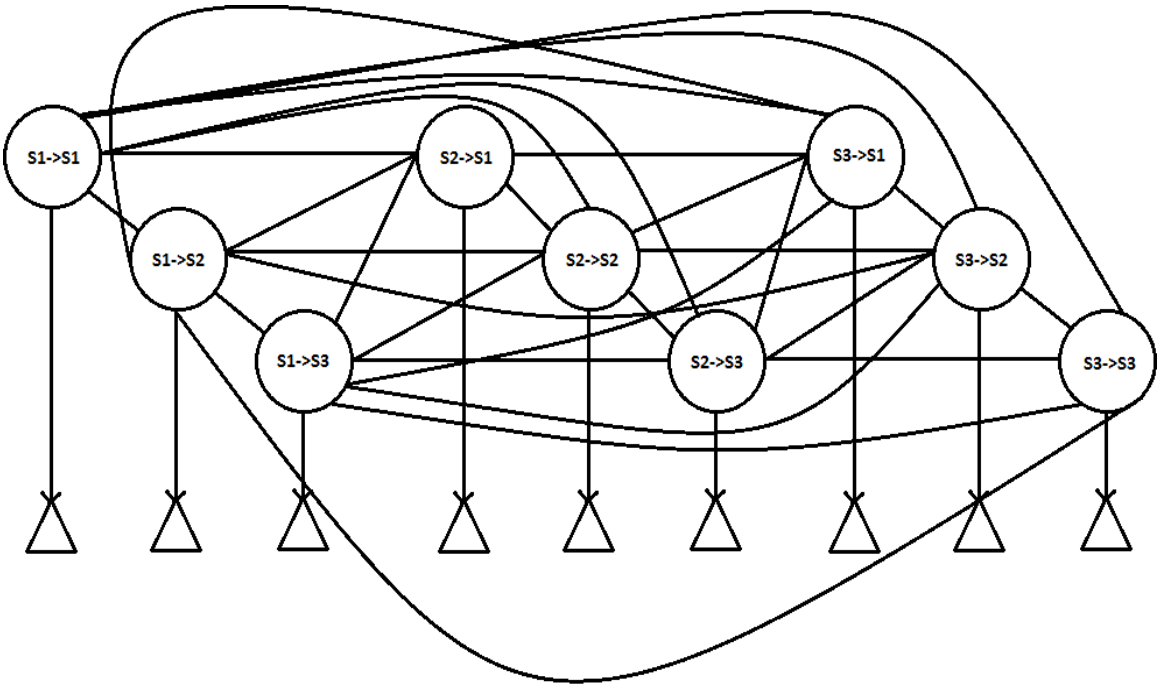

Figure 16. A graphical representation of a bigram extension to a three hidden state unigram HMM (e.g the one in Figure (12)). Note that there are 3^2 hidden states in this graph, because all possible "two-state" transitions in the unigram model have been folded into their own states. Note that the links between hidden states are undirected, but the links between hidden states and observed states are directed. Although it is not depicted, the number of observed states would also increase. If this model was used for PoS tagging, then the hidden states would correspond to every pair of PoS tags and the observed states would correspond to every observed pair of adjacent words in the corpus.

10.10.2 Manipulations of Underlying Mathematics

Although it is possible to extend the modeling power of HMMs to create techniques that are consistent with the basic mathematical assumptions of standard HMMs, some extensions to Markov Models are designed to assume different properties of the stochastic processes to be modeled. These types of HMMs usually fall into the category of those less applicable to NLP (aside from application to speech perception tasks). For instance, there are HMMs that exhibit a variable state transition rate (Sen, Maglaris, Kikli, & Anastassiou, 1989) or time steps of variable duration. Another minimally relevant extension to Markov Models, the non-homogenous Markov Model,

involves defining a time-variable transition matrix (Rajagopalan, Lall, & Tarboton, 1996).

However, it must be noted that these more involved models not only are less widely adopted by NLP researchers, but also are not to be found in the set of top-performing techniques for many NLP problems. Considerations of computational tractability limit the use of such models in most NLP applications, because the efficient and simple algorithms for training basic HMMs often need to be modified in some way to handle the demands of more complicated models. Hidden Semi-Markov Models (HSMMs) (Yu, 2010)serve as an illuminating example of this difficulty. A Semi-Markov process is one that does not globally impose the Markov Assumption. Instead, Semi-Markov processes become Markovian only at the times of state transitions. State time transitions can be considered as a separate process form the state transitions themselves. Because the Markov Assumption does not hold over the entire model, inference is generally intractable, although approximate methods exist (Moura & Droguett, 2008).

10.10.3 Increasing Model Structure

There are a large number of extensions to the basic configuration HMM models that involve manipulations of model topography to allow complex processes to be more adequately represented. One such approach is to divide the process being modeled into a combination of several processes. Factorized (or Factorial) HMMs (Ghahramani & Jordan, 1997) are an example of this, which are related to Mixed Memory Markov Models (Saul & Jordan, 1999). These model variations are used to break down stochastic processes with complex structure into mixtures of simpler processes. Evaluations of this variety of HMMs reveal that they can outperform HMMs on some problems and are generally capable of representing types of structural patterns in data that HMMs cannot. However, exact inference for Factorized HMMs is intractable. Some attempts to apply Factorized HMMs to NLP tasks have been made, namely for tagging tasks (Ramanujam & Baldridge, 2009). Factorized HMMs have proven to outperform HMMs on the supertagging[47] task.

[47] For a description of supertagging, which is a more complex form of PoS tagging, see: (Bangalore & Joshi, 1999)

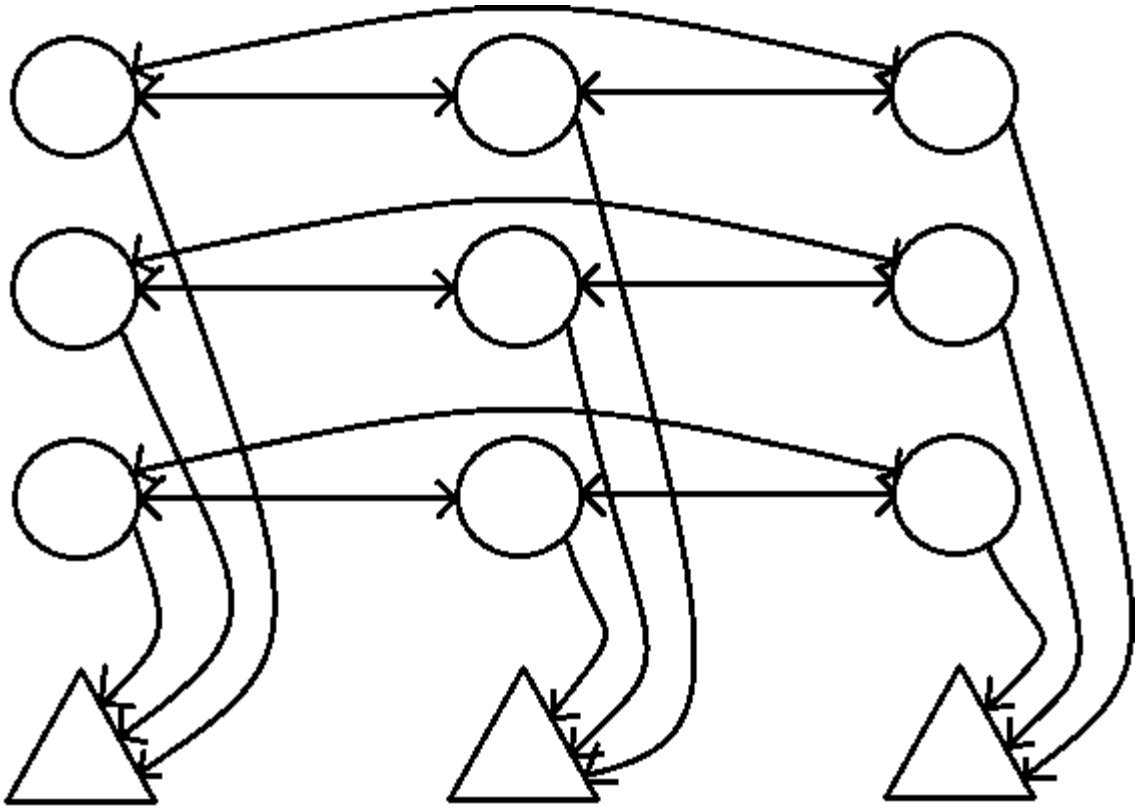

Figure 17. This is a graphical representation of a Factorial HMM, where the hidden states are represented as circles and the set of observed sets are represented as triangles. Unlike simple HMMs, note that there are multiple parallel hidden stochastic processes that each condition the observed state at each time step. Within each hidden process, all state transitions are allowed. Note that there are not transitions between hidden process states, only between hidden states and observed states.

An even more complex configuration modification of HMMs results in the production of Hierarchical HMMs (Fine, Singer, & Tishby, 1998). As depicted in Figure 18, a Hierarchical HMM is like a Factorized HMM in that there is more than one stochastic process considered by the model. However, as compared to Factorized HMMs, additional structure is imposed on the decomposed stochastic process. Hierarchical HMMs essentially consider stochastic processes to be comprised of several different levels of abstraction, which correspond to different time scales in the behavior of phenomena. Originally, these models were used for modeling human activities, which can be cast as sequences of sequences of behavior that unfold over different time scales. Since natural languages are arranged in hierarchical structures—from phonemes in

morphemes to morphemes in words to words in phrases to phrases in sentences to sentences in paragraphs to paragraphs in larger works—it follows that a Hierarchical HMM would be more suited to modeling the complexities of natural languages than other configurations of this family of models. For instance, one could imagine setting up a Hierarchical HMM to model natural languages in the following way. The observable states would be words, as with a standard HMM. The first level of hidden nodes would correspond to types of phrases in the language that is being modeled. The second level of hidden nodes would correspond to sentences and so forth. However, there are several issues with kind of set up.

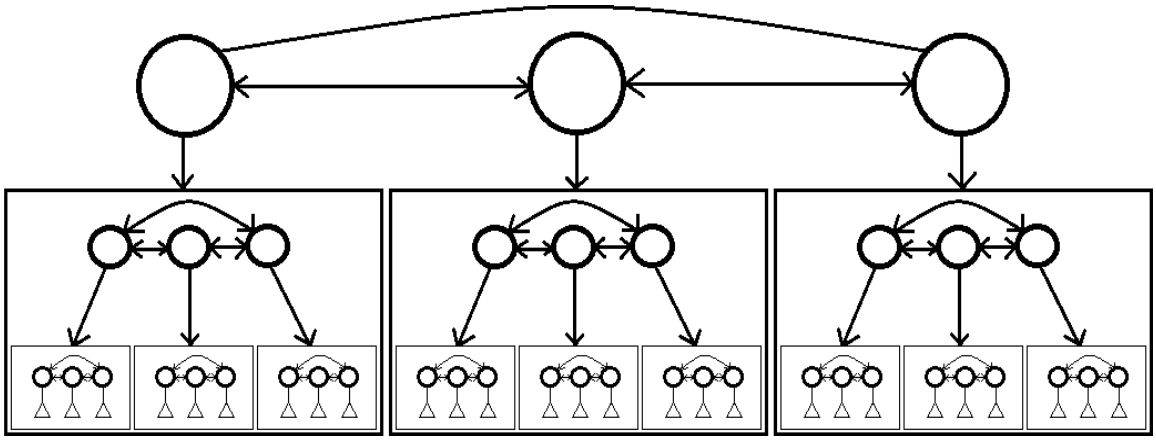

Figure 18. A graphical representation of a Hierarchical HMM, where hidden states are represented as circles, intermediate observed states are represented as rectangles, and ground observed states are represented as triangles. Note that higher level processes in the model have observed states that are composed of lower level hierarchical processes. The lowest level models have observed states that are congruous to simple HMM observed states.

Some languages, such as English have a huge number of types of phrases, when all of the relative parameters are taken into consideration: there are different kinds of NP depending on whether or not there is a quantifier involved, whether the head of the phrase is a mass or count noun, male, female, un-gendered, and a number of other qualities. Special linguistic constructions (Goldberg A. E., 2003) also would be challenging for this type of scheme, because they involve phrasal arrangements that do not follow from the main grammatical rules of a language. An example of a construction in English is: *the more X, the more Y*, which can be interpreted to mean *increasing the level of X is known*

to increase the level of Y. Elided elements (Lappin, 1996) in language would also be difficult for this type of approach to accommodate, because the involve words that are missing in the corpus. For example, the sentence *He went to the store and is returning home* is an elided form of *He went to the store and he is returning home* that has the second incidence of *he* removed.

It is difficult to see how any kind of HMM, hierarchical or not, would be able to recover this kind of information without explicit knowledge about the semantics of the language being models. Furthermore, the only way to ensure these different kinds of structural states are represented is to allocate a sufficient amount of hidden states for each level in the hierarchy. There is no direct provision to incorporate knowledge about the language in these models, so the best a modeler can do is to specify a large enough number of hidden states at each level to approximate a number of categories that would enable some reasonable amount of coverage. Also, there is no ability for these models to expand their state space during inference (rather than the trivial use of smoothing techniques to simulate the processing of untrained data patterns). Having to represent all of these structural statues not only greatly increases the complexity of the model, but creates a need for a much larger amount of data to prevent undertraining.

Because of the large state space and the computational demands of training a Hierarchical HMM, it is intractable to execute exact learning algorithms to train this type of model. Instead, it is necessary to perform approximate training through simpler estimations of likelihood (and perhaps other measurements as well). This results in a corresponding decrease in model accuracy. Furthermore, as with all other attempts to extract more structure from corpora with HMM-based models, there are issues with needing to model the entire join distribution of hidden and observed variables. This can be difficult to do while still attempting to learn complex patterns from data. Therefore, the various levels of an HMM would be simplistic themselves and would not be able to easily take certain long-distance dependencies that are found in natural language into account. Adding additional structure to an HMM's representation of a stochastic process is not sufficient to accommodate all features of languages beyond those found in regular expressions.

Additionally, the different levels of Hierarchical HMMs represent different time scales and not, perhaps, other types of higher-order organizations. For strict word-order languages, such as English, this would probably not pose many direct problems (although poetical, dialectal, or antique forms of English, such as Elizabethan English), but there languages with much freer word order, such as Russian (Kallestinova, 2007) or Warlpiri (Nash, 1980). In free word order languages, phrases also exist, but they may not be organized contiguously in time. Some kind of provision would have to be made so that the groupings of words could be trained via inflections or other grouping cues that would not be explicitly labeling in a corpus for a language, such as, English that does not heavily rely on such features. So, despite the fact that Hierarchical HMMs are capable of modeling more complex types of stochastic processes and that they seem to be more representative of the structure of natural languages, they still are not a panacea for the problems of statistical NLP.

10.10.4 Increasing Complexity of Modeled Information

The states of HMMs can be structured in a much more sophisticated way than a simple random variable. As will be indicated below, it is possible to implement discriminative classifiers inside of the states of HMMs. Beyond that, it is possible to incorporate language-specific elements in the hidden states of an HMM. For instance, a semantic hierarchy has been merged with a HMM to enable this kind of approach to solve semantically-driven NLP problems (Abney & Light, 1999). Semantic Hierarchies provide information about the types of entities and the structure of ontological hierarchies that can be useful for determining word sense from the surrounding context. Relational HMMs (Anderson, Domingos, & Weld, 2002) are capable of encoding arbitrary propositional relations, not only categorical relationships Logical HMMs (Kersting, De Raedt, & Raiko, 2006) are a further generalization of Relational HMMs that enable variable binding and unification. This kind of technique allows the different states and transitions of an HMM to be coded to correspond to expressions with argumentation and actions on those items. Additionally, Logical HMM enable the creation of procedures that require some notion of internal logical structures to understand, so it is conceivable that this type of approach could learn different parsing sequences in a language if they

were represented as chains of function-like commands. A basic conceptual depiction of a Logical HMM can be found in Figure 19 below.

Logical HMMs are capable of handling tree-like language structures and are thus nearly equivalent to the models used to process Context-Free Grammars (Kersting, De Raedt, & Raiko, 2006), although they have some favorable computational properties in comparison with standard Context-Free processors. While this may seem to be a major advance in the technology available for NLP research, there are sections below that describe why even Context-Free level complexity is insufficient for human-level NLP. At last, a combination of Logical HMMs and Hierarchical HMMs has resulted in the culmination of HMM-like formalisms: Logical Hierarchical HMMs (Natarajan, Bui, Tadepalli, Kersting, & Wong 2008). This enables the learning of stochastic processes that involve semantically related states that are also arranged in hierarchical patterns, such as the behavior of users of computer systems. Ultimately, whatever increases in efficiency these approaches offer are rendered useless because they cannot model fundamental aspects of human languages that are required to completely solve the NLP problem. Like the HMMs with more complex model structure, these approaches suffer from a reduction in tractability due to their increased state size. Learning for these approaches tends to be restricted to approximate methods, which can decrease performance significantly, especially since hardly any HMM-based approaches are capable of self-modifying in a self-guided, iterative process beyond initial training.

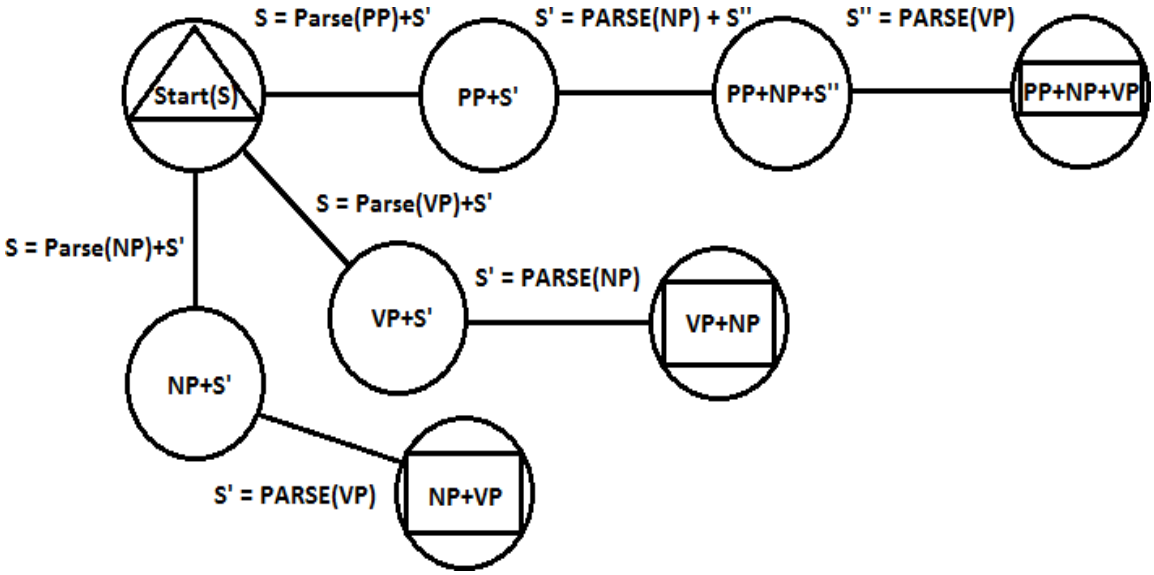

Figure 19. A graphical representation of a Logical HMM that might hypothetically be used to simulate syntactic parsing. Each state represents some stage in the parsing of a sentence. Each link represents a new parsing action that can be taken from a particular parse state. States with a triangle are initial states and states with a rectangle are end states.

10.10.5 Summary of Extensions to HMMs

Overall, the many extensions to HMMs do little to address the fundamental limitations of HMMs. While these variations may ameliorate some specific issues and may enable hand-optimization to certain types of problems, it is clear that the HMM framework is far too limited to support the goal of human-level NLP. Going beyond HMMs in the sections below, a number of recurring themes can be found. Like HMMs, knowledge-lean approaches suffer from being limited to extremely simple constraints, a lack of self-modification and generalizability, and a reliance on human operators to correct and modify any problems that arise.

10.11 Overview of the Markov Model Framework

Although there are an immense number of ways that the basic Markov Model framework can be modified to handle a large number of diverse classes of problems, there is a high degree of commonality between these methods from the perspective of gauging their effectiveness in providing the foundation for a system geared towards achieving human-level NLP performance. The fact that these common themes exist

serves as a convenient way to understand why Markov Modeling has been used by the NLP community, despite the fact that it has a number of serious shortcomings.

As a brief recap, recall that Markov Models make use of some version of the Markov Assumption, which enforces a high degree of locality in the causal structure of model state transitions. While this assumption not only seems to appropriately capture the behavioral evolution of many kinds of processes over time, it also enables the implementation of tractable solutions to otherwise computationally-intractable problems. This reduction in complexity, however, comes at the price of limiting the expressivity of Markov Models and therefore also decreases their ability to model complex phenomena. Indeed, a basic HMM is equivalent to a finite state transducer, which is the corresponding kind of automaton for Regular Languages. To overcome this limitation, new extensions to the Markov Framework are developed to handle the increased demands of more complex processes on a case-by-case basis.

10.11.1 Positive Qualities of Markov Models

Perhaps the most salient and universal positive quality of the Markov Modeling framework is how extremely easy and inexpensive it is to apply Markov Models to a huge variety of problems. Indeed, this quality is probably one of the chief reasons that Markov Models have stood the test of time and have so far been utilized in mathematical modeling for over 100 years. To obtain an understanding of how diverse the Markov Modeling Framework truly is, consider the Wolfram Demonstrations Project website on Markov Model Applications: (Wolfram Research, 2013). Although these modeling applications do not necessarily deal directly with the NLP research problem, it is clear that Markov Models are still very much utilized by all branches of scientific modeling. Because of this popularity, there are many available toolsets and/or well-known algorithms for programming Markov Models that modelers can exploit.

Furthermore, the HMM framework is extremely malleable and can be adapted to handle data of high complexity through the use of one or more of its various extensions: Factorized HMMs, Hierarchical HMMs, Logical HMMs, etc. It is possible to approximate the level of expressivity of Context-Free Grammars using appropriate extensions to the Markov Modeling Framework. This quality enables Markov Models to

be used to model a wide variety of NLP phenomena while still allowing researchers to operate within the same general mathematical framework.

In light of these benefits of Markov Models, it is not surprising that they are so popular for NLP programming. However, it is also the case that Markov Models, perhaps after some modulation, are essentially as accurate as any other method on many NLP tasks. The cost-benefit ratio for using Markov Models is at least as high as that of many other approaches.

10.11.2 Negative Qualities of Markov Models

Although there are a number of reasons why Markov Models have become such a popular framework for NLP applications, there are also a large number of factors that cause Markov Models to be suboptimal for this task. For instance, despite being couched in the knowledge-lean, data-driven modeling paradigm, Markov Models are fundamentally non-adaptable. First, Markov Models are trained to very specific and rigid patterns in corpora. They can be trained on unstructured data, but this almost always leads to unacceptably low accuracy. As a consequence, Markov Models are quite poor at generalizing to data environments that differ from the ones in which they were trained. Furthermore, the training algorithms used with Markov Models themselves are likely to lead to overfitting, especially when there is relatively little data in proportion to the size of the model (i.e. larger, more complex models require more data to avoid overfitting). Due to these factors, even for tasks that Markov Models are extremely well-suited for solving, performance is generally low when any modifications to the initial training assumptions are made. HMMs in particular were originally deployed (in the NLP arena) to handle speech recognition tasks, and, by some measurement, this area is where they have enjoyed the greatest amount of success. However, when an HMM trained to perform speaker independent phoneme recognition can be as high as 23.1% between different speakers from the BREF[48] corpus (Lamel & Gauvain, 1993).

Furthermore, the fact that basic Markov Models are only as expressive as regular languages makes them somewhat poorly fitted to accommodate the complexity of NLP problem. In particular, natural languages are known to be at least context-free in

[48] C.f. (Lamel & Gauvain, BREF, a Large Vocabulary Spoken Corpus for French, 1991)

complexity. While it is possible to eke out higher complexity from Markov Models by increasing the model state space in some way, it turns out that these more complex varieties of Markov Models are computationally expensive. The more complex a Markov Model becomes, the training and inference algorithms that are used to support them must become increasingly more approximate and hence are considerably less accurate. Additionally, these larger models require vastly larger corpora to properly train, which is a liability since annotated NLP corpora still need to be created by hand. A final downside of utilizing the extensions to the Markov Modeling framework is that the decision to use and the design of such models can only be completed through direct human intervention. Recall that one of the major rationales for using knowledge-lean frameworks in the first place was to enable models to be more automatic and to reduce the presence of humans in the inference loop.

Aside from the fact that Markov Models are not naturally expressive enough to accommodate the basic structure of language, there is an additional liability stemming from the fact that HMMs, in particular, are generative models. Generative models must model the entire joint distribution between observed and hidden states; thus they must contain information about how to generate observed states from hidden states, as well as having information about how to generate hidden states from observed states. This requirement makes it difficult for HMMs to incorporate much complexity at all into their design, let alone complex, relational, knowledge-rich structures for encoding world knowledge. Such knowledge is almost certainly required for human-level NLP performance.

Despite all of these shortcomings, it is tempting to claim that, nevertheless, there is very little that the Markov Modeling framework cannot accomplish by some amount of modification of its basic configuration. This concern is actually one of the common themes encountered in the course of researching deeply into the mechanics of knowledge-lean modeling formalisms and will be seen to also be true of most other frameworks. Yet, the theoretical, potential capability to handle a particular kind of problem does not imply that reasonably efficient implementation of the necessary models is possible. Indeed, as more complex information is folded into a Markov Model, the resulting state

space over which learning and inference must occur explodes to a degree that the ingenious algorithms developed for basic Markov Models fail to scale up.

10.11.3 State Space Explosion

Of all of the drawbacks of the Markov Modeling framework, the one worth the most consideration here is the issue of the state space explosions that occur whenever Markov Model variants are created to address the fundamental limitations on expressivity inherent in this methodology. Consider what is perhaps the most basic kind of extension to an HMM, namely the practice of increasing the model memory so that more than just the previous hidden state influences the current hidden state of the model. By taking into account what the PoS tags of the two previous words (or even the three or fourth previous words) were, the resulting model requires far greater amounts of memory and processing speed. In practice, most HMM-based models are limited to trigrams (i.e. three words are taken into consideration). Trigram models perform well on the PoS task, but they fail to meet the mark that is required for human-level NLP parsing.

One extremely telling point in the history of the knowledge-lean paradigm is worth repeating—these methods were only taken seriously once computers became powerful enough to support knowledge-lean models that could perform about as well as knowledge-rich models. In the end, knowledge-lean models are easier to design, deploy, and maintain than knowledge-rich models. Even though it is clear that knowledge-lean methods will only be as powerful as available computing power allows, this does not deter NLP and AI researchers from investing almost exclusively in the knowledge-lean paradigm. Yet, no realistic amount of increase in computer power will enable fundamentally mismatched algorithms to achieve human-level performance; better modeling platforms must also be created into order to accomplish this goal.

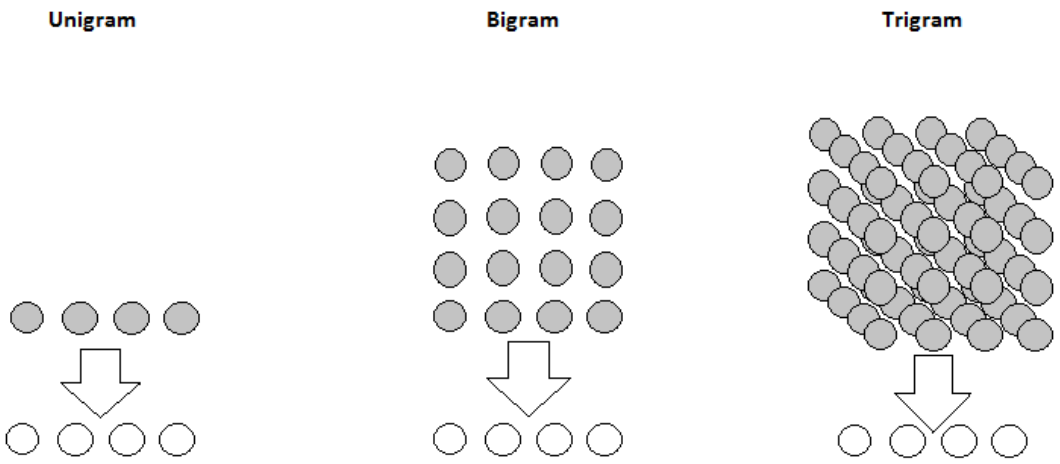

Figure 20. State space explosion for increasingly complex HMMs. Shaded circles represent hidden state nodes. Open circles represent observable state nodes.

11. Maximum Entropy (ME) Modeling

One of the main shortcomings of HMMs, as discussed above, is its inherent limitation to learning highly localized patterns in data. Since natural language is known to contain many long-distance dependencies that, at least on the surface level, do not appear to be reconcilable with locality principles (Zushi, 2001), it would seem that HMMs are a poor choice for creating human-level NLP systems. Higher-order HMMs, and, to some extent, the more complicated formulations of hidden states, provide a limited means to extend the scope of patterns that are recognizable by such methods. However, since these techniques greatly increase the state space (and search space) of HMMs, there are strict practical limits on how far they can be extended. Another problem with the HMM framework is that it belongs to the generative family of statistical models. This has the effect of limiting the types of patterns or constraints that can be obtained from data. Essentially, all of the structure to be learned by an HMM has to be explicitly annotated in the training data. HMM taggers use word co-occurrence data because this co-occurrence structure is provided automatically through the text itself. Producing more complicated data would not only require a much larger hidden state space, but would also require the creation of large-scale, structure-rich annotated corpora. Both of those requirements are intractable in practice due to limitations on computational and human resources, respectively.

In order to address these specific issues, researchers developed techniques that are capable of learning constraints that can take a much greater range of forms than lexical co-occurrence data. The Maximum Entropy Modeling framework is one of the most widely used and best understood of these approaches. The mathematics of ME models are based on logistical regression (Nerlove & Press, 1973), which is also similar to the algorithms used for decision tree induction (Quinlan, 1986). A concise introduction to the mathematical theory that underlies the operation of ME models can be found in: (Ratnaparkhi, 1997). ME models are a type of discriminative classifier—they learn how to map incoming data into one element of a set of classifications. As opposed to generative models like HMMs, discriminative models do not need to model the full joint distribution between the input and output random variables. An extensive study of the application of ME models to various NLP tasks is described in (Ratnaparkhi, 1998). For

the purposes of this document, a simplified, conceptual explanation of ME models is provided, specifically as they can be applied to the PoS tagging problem. ME models rely on sets of constraints that are used to match patterns in data in an attempt to determine how larger, more general structures in the data arise from smaller individual features.

To see how more expressive constraints are able to improve upon simple lexical co-occurrence patterns, it is easiest to consider a concrete example. Although the task of PoS tagging can be solved quite accurately with higher-order HMMs that are configured to learn over trigrams, there is room for improvement, especially if such improvements can be made without making individual knowledge-engineered corrections to the model. According to (McCallum, Freitag, & Pereira, 2000), the representation of language in a HMM is equivalent to a multinomial distribution over the set of words that could be encountered by the model. However, there are a number of useful features for determining the PoS of a word other than the identity of a few words around it.

By studying the patterns of word use in a language it is possible to establish what general types of patterns are important for learning a particular task. These patterns can be broken down into simple elements call features, which can be compiled together in various permutations to form constraints. Some of these features are: if the word is capitalized, if the word ends in a particular suffix, or if the word comes before or after some other word with a particular PoS tag. An illustration of how these types of constraints could be represented is provided in: (Ratnaparkhi, 1996).

There are two aspects of defining these constraints. The first is to determine the available "history" for forming constraints, which is defined relative to the current word to be tagged. This "history" is all of the raw material that can be extracted from the training data by the model, so in some cases, the "history" involves information "in front of" the current word[49]. Note that, for the purposes of this example, the words in a

[49] Having insight into the future content of a sentence might seem unrealistic, but it is possible that the human cognitive system integrates information in a kind of sliding buffer or window. For instance, although some processing might begin as soon as a word is heard—this processing may be conditioned on previous word--the system may not finalize its conclusions until several subsequent words (or silence) have been perceived. Psychometric studies of eye-movements during reading seem to support this claim (Engbert, Longtin, & Kliegl, 2002). Direct evidence for a neural implementation of speech signal integration over time has been found, albeit on a smaller timescale than would be required for most multi-

sentence (and the PoS tags on those words) are seen as a time-ordered sequence that begins and ends at the sentence boundaries. There are some constraints on the elements of the history. Primarily, the elements in the history have to be available both during training and evaluation. Assuming that the evaluation data is unannotated places some obvious limits on what can be included in the history. For instance, it is not possible to include PoS tags after the current word under consideration, as those would be unavailable during evaluation processing. PoS tags on words before the current word would be available during training and evaluation, if the system itself was designed to compute prior PoS tags by the time some decision has to be made regarding the current word. Additionally, there are constraints on the size of the history, which influences the number of constraints that can be generated by the model. As more constraints are programmed into a model, the size of the problem space increases in a way similar to that of increasing the order of HMMs.

With the limitations on the possible raw material for features and constraints understood, it is possible to illustrate what these histories are for different types of models. Assuming that the training texts are annotated with PoS tags, the possible range of history elements would be: $\{w_{t-n}, w_{t-n-1}, \dots, w_{t-1}, w_t, w_{t+1}, \dots, w_{t+m}, tag_{t-n}, tag_{t-n-1}, \dots, t_{t-1}\}$, where w_i represents a word in the current sentence at position i, t_i represents the tag of a word in the current sentence at position i, and time t is the current position in the sentence. Therefore, the history consists of all of the words surrounding the current word in any direction, as well as all of the tags on words before the current word. In theory, a maximum entropy model could utilize this history for arbitrarily large values of n, m, and x. Only computational limitations restrict the size of the history, although it is probably the case that extending the history indefinitely would lead so some asymptotic limit on accuracy. By comparison, the history of a trigram HMM tagger would be restricted to $\{w_{t-2}, w_{t-1}, w_t\}$.

The ME PoS tagger described in (Ratnaparkhi, 1996) uses the history: $\{w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2}, tag_{t-2}, tag_{t-1}\}$, which seems to be motivated by the idea

---

word utterances (Obleser & Kotz, 2011). Models of situated natural language comprehension have also relied on a processing-window based approach (Bordes, Usunier, Collobert, & Weston, 2010).

that a trigram HMM tagger provides a high baseline accuracy to improve upon. However, further development of ME PoS taggers have revealed that there is little gained by using words or tags that are displaced by more than one index from the current word (Toutanova & Manning, 2000). These models, which rely on more linguistically-motivated constraints, proved that a history of $\{w_{t-1}, w_t, w_{t+1}, tag_{t-1}\}$ is sufficient to obtain state-of-the-art accuracy.

Once the scope of the history is established, it is possible to define features over the types of elements in the history. In general, a feature is defined as a binary function that takes a set of history elements and returns either 0 or 1 depending on whether or not a condition over the input elements holds. The formal notation to describe a feature is:

$$f_i(h_1, \ldots, h_n) = \begin{cases} 1 & if\ CONDITION(h_1, \ldots, h_n) \\ 0 & else \end{cases}$$

Figure 21. Basic feature structure for an ME Model.

For an HMM, the only feature is word identity defined over the number of words it uses to condition the PoS tag of the current word. In the case of a trigram HMM, there are three words that are used: the current word, the preceding word, and the word preceding the preceding word. Note that the words other than the current word may be blank (e.g. if the current word is the first or second word of a sentence). An example feature for a trigram HMM would look like:

$$f_i(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & if\ w_{t-2} = null\ \text{^}\ w_{t-1} = \text{the}\ \text{^}\ w_t = \text{man} \\ 0 & else \end{cases}$$

Figure 22. A feature for a trigram HMM in ME Model notation.

Here, the current word is at time index t. This feature is activated under the condition that the word *the* is the first word of a sentence and the word *man* is the second word of the sentence. So, for the sentence *the man was tall*, this feature would apply, but it would not apply to the sentence *a car went by*. Note that this feature could very well be used by a ME model, but ME models are capable of more complex forms of constraint.

Unlike a trigram HMM, a PoS tagging ME model's features can take previous tags and word substrings into account. Such a feature might be:

$$f_i(tag_{t-1}, w_t) = \begin{cases} 1 & if\ tag_{t-1} = \text{DET} \wedge EndsWith(w_t, \text{ing}) \\ 0 & else \end{cases}$$

Figure 23. A feature for an ME Model that is beyond the expressivity of standard HMMs.

This feature would apply to a sentence that contained a word in any position that ended in *-ing* and was preceded by a word marked as a determiner. Thus, this would hold for the sentence *it was the waiting that upset her*, but not for the sentence *the time has come*. The information about the tag could come from an annotation in the training corpus or from the previous action of the model itself. It is clear that these additional features have the potential to boost the performance of ME models in comparison with HMMs.

One point to make clear is why HMMs do not take more sophisticated features into account. The main reason behind this is that HMMs are generative models and therefore must consider the entire joint distribution between hidden states and observed states. Although it might be possible to determine the conditional probability that a word ending in *–ing* is a particular type of speech, but it can be harder to estimate the reverse probabilities, for instance that a particular type of speech will generate an ending of *–ing*. Having to consider these additional probabilities is expensive and restricts generative models from learning constraints for which there is little or no evidence about how they generate the data. Moreover, for the task of PoS tagging in particular, it is unnecessary to model how a PoS tag generates individual words. Although an HMM might be able to represent the same kinds of constraints that are possible in ME models, this would require the definition of a much larger number of hidden states, which in turn would require a much larger amount of training data to avoid poor conditioning of the model's parameters.

It is clear that there are a large number of possible constraints for a ME model. This number is prohibitively large to including all such constraints in any given model. Moreover, it is unlikely that all constraints will significantly contribute to the overall results, depending on whether or not they capture important patterns in the data. For

example, a constraint that involves tags and word forms rather than individual words would probably have more importance than a constraint that involved a sequence of infrequently used words. To know which constraints to include a priori is an extremely difficult problem. As was mentioned in the section about knowledge-engineered approaches to NLP, some researchers have performed a thorough manual analysis of the patterns in languages, namely the Constraint Grammar approach that hand-coded thousands of PoS tagging rules. Not only is this approach extremely time-consuming, it is also intractable to find every single possible rule that might be encountered in a language. This is clear if the large number of idiomatic cases, grammatical nuances of individual dialects, and corrigible errors that appear in natural language use. Furthermore, updating a hand-coded system when new rules are discovered is potentially a time consuming process if the rule used are weighted by hand or arranged in a decision hierarchy.

To circumvent the problem of having to manually create rules, ME modelers employ what is known as a feature template. A feature template is an abstracted form of a feature that can be instantiated to create a large number of specific grounded features. For instance, a feature template might generate a feature that captures the identity of the two words that precede the current word and the current word:

$$ft_i(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & if\ w_{t-2} = \text{X} \wedge w_{t-1} = \text{Y} \wedge w_t = \text{Z} \\ 0 & else \end{cases}$$

Figure 24. A feature template for an ME Model.

In the feature template above, X, Y, and Z are variables that correspond to any word that appears in the text being modeled. Note that a trigram HMM could be thought of as having exactly one feature template that is defined in the same way as Figure 24, which would take all of the three-length lexical co-occurrences in a text into account.

The use of feature templates requires ME models to make at least two passes over a training text, whereas HMM-style models only require one pass. First, an ME model must do an initial pass to instantiate the individual features from the defined feature templates. If the feature template defined in Figure 24 above is applied to the following text: *this is a test*, then the following instantiated features would be created:

$$f_1(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & f\ w_{t-2} = \text{null} \wedge w_{t-1} = \text{null} \wedge w_t = \text{this} \\ 0 & else \end{cases}$$

$$f_2(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & f\ w_{t-2} = \text{null} \wedge w_{t-1} = \text{this} \wedge w_t = \text{is} \\ 0 & else \end{cases}$$

$$f_3(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & f\ w_{t-2} = \text{this} \wedge w_{t-1} = \text{is} \wedge w_t = \text{a} \\ 0 & else \end{cases}$$

$$f_4(w_{t-2}, w_{t-1}, w_t) = \begin{cases} 1 & f\ w_{t-2} = \text{is} \wedge w_{t-1} = \text{a} \wedge w_t = \text{test} \\ 0 & else \end{cases}$$

Figure 25. Populated feature templates for an ME Model.

After the first pass through the training data has instantiated all of the features, it is necessary to determine how these individual features contribute to the overall structure of the distribution being modeled. Intuitively, these features are of differing importance for characterizing the distribution of PoS tags in a text. This is the parameter estimation phase of the model creation process. The model features are assigned weights so that they best represent the incidence characteristics of the features as they appear in the training data. A method called Iterative Scaling (Darroch & Ratcliff, 1972)[50] is employed to determine the optimal weights of the features given a particular training text. Essentially, this method is based on detecting the divergence of the distribution described by the current set of feature weights and then updating the feature weights so that they better reflect the target distribution. Once convergence has been obtained, the iteration process is stopped and the weights at that point are close to an optimal characterization of the distribution of features in the training data. Very large models require Monte Carlo algorithms to approximate the learning of optimal feature weights. Note that such methods are possible because the ME framework makes no assumptions about the independence of features.

In more complex models, the atomic features above can be combined into compound features, which consist of a linear combination of two atomic features or an atomic feature and a compound feature (Della Pietra, Della Pietra, & Lafferty, 1997). These feature combinations are used to create more sophisticated constraints that can

[50] A generalization of Iterative Scaling, Improved Iterative Scaling (Berger A. , 1997), reduces restrictions on the configurations of features that are allowed in ME models.

improve how well a model is fit to a particular data distribution. Although generating a larger feature space through feature combination can be useful, it quickly becomes in tractable because of the complexity of the parameter estimation algorithm. Therefore, it is often the case that it is useful to include only the features that significantly contribute to the fitness of the model. This phase of model creation is called feature selection. Efficient algorithms for feature selection have been developed (Zhou, Weng, Wu, & Schmidt, 2003), which has proven to be an important advance for the practical application of ME models. To gauge how informative a particular feature is, the maximum "gain" that the feature contributes to the fitness of the model can be measured by exploring the impact that the current feature has on the distribution characterized by the model. This gain is measured by examining the effect of assigning different weights for the current feature from a selection of values in the range of possible weights. It can be shown that these calculations form the dual of a numerical method that is guaranteed to converge to a global optimum (Berger, Della Pietra, & Della Pietra, 1996) under reasonable assumptions.

Finally, ME models have a provision for handling the variance in the target distribution that is not explained by the features. Inevitably there will be some information in the training corpus that cannot be captured by the constraints in the model. For instance, if commonsense world knowledge is the strongest influence on the PoS of a word, features such as the ones presented above would not faithful capture this relationship. Consider the PoS-ambiguous sentence: *The duchess was entertaining last night*. Here *entertaining* could be either an adjective or a verb participle. However, in the sentence: *The duchess was entertaining last night, but she ended up boring her guests*, it is much more likely that *entertaining* is describing what the duchess was doing (i.e. hosting a diner) rather than an attribute about the duchess (i.e. that she was an entertaining person). Given a two-word "history" window, the additional words added to disambiguate the PoS of *entertaining* would have no impact on the decisions of such a model. Even if the history window was expanded, the form of the constraints given above would not suffice to capture how the additional words influence the PoS of *entertaining* through semantics.

To account for such gaps in the modeled distribution, it is necessary to have a "fallback" strategy so that unforeseen patterns in the data can nevertheless be processed. ME models use the principle of Ockham's Razor to perform this function: provide the simplest explanation for what is unknown (Jurafsky & Martin, 2000). It turns out that the simplest explanation is to assume that the gaps in the model are equally likely to be filled by each of the possible classification outputs of the model. An example of a gap would be if a word seen in a training corpus had only one PoS tag associated with it, say NN, then all of the other PoS tags probabilities for that word would be set to 0. Ockham's Razor also implies that, all things being equal, the classification probabilities should be as evenly spread to each class as possible for each of the words in the corpus. By analogy, this serves a similar purpose as smoothing does for HMMs. It is impossible both to produce a corpus large enough to contain all possible constructions in a language and to create large enough constraints to cope with the long-distance dependencies that might occur in NLP[51].

11.1 Direct Applications of ME Models

Even though traditional ME models are not inherently capable of modeling sequential processes in the same sense that HMMs can[52], they have been used to model some NLP tasks that can be represented in a static way. ME models can be deployed to solve problems at various levels of complexity. For instance some selected ME applications are as follows: 1.) Identifying the boundaries of sentences (Reynar & Ratnaparkhi, 1997), 2.) PoS tagging (Ratnaparkhi, 1996), 3.) Prepositional phrase attachment (Ratnaparkhi, 1998), 4.) Phrase chunking (Skut & Brants, 1998), 5.) Parsing (Ratnaparkhi, 1997), and 6.) Information extraction (Chieu & Ng, 2002). As a comparison with HMMs, the ME PoS tagger model in the above list is capable of 96.31% accuracy on all words without the use of a tag dictionary. Recall that HMM PoS taggers achieve between 97% and 98% accuracy, so the performance of ME models on this tasks is comparable.

---

[51] Some languages have structures that are necessarily long-distance (or require complex analyses to cast them as local features): (Dalrymple, 2011).
[52] Because they do not have the structure required to model sequences of data over time.

It is also possible to create ME models that aid in solving certain types of specific NLP problems that other approaches find difficult. For instance, the paper (Alumae & Kurimo, 2010) addresses the problem of adapting a model trained on a corpus based on written language to corpora that are based on transcripts of various styles of spoken language. Differences between the styles of written and spoken language cause unadapted models to perform poorly on this task. Presumably, this work also addresses the problem of data scarcity due to insufficient corpora sizes of particular speech styles, since the main model is based on training from a large written text-based corpus.

Language specific applications have also been identified for the ME modeling framework. For instance, some NLP problems are much more difficult in particular languages in comparison with others. Segmentation is a prime example of such a problem. In (Fung, Ngai, Yang, & Chen, 2004), an ME framework is used to solve the difficult problem of Chinese character segmentation. Previous models for parsing Chinese heretofore assumed that the segmentation problem would be solved a priori rather than computed as part of the parse.

11.2 Maximum Entropy Hidden Markov Models (MEMM)

Despite the fact that ME models can be used for a large number of tasks, it turns out that some of the properties of HMMs are necessary for obtaining good results on NLP tasks that are inherently sequential in nature. To accommodate this fact, researchers have developed a way to combine ME classifiers within an HMM-style network of hidden nodes. One of the major shortcomings of generative models, such as HMMs, is their limited ability to incorporate any complex information source beyond the most basic features. As mentioned above, lexical co-occurrence patterns are the extent that unstructured HMMs can handle; structured HMMs are generally intractable, which forces them to rely on approximate inference algorithms and leaves them vulnerable to overtraining. Worse, the actual constraints in generative models are completely "unpacked" into a large number of co-occurrence constraints. ME Models directly address these issues by allowing explicit constraints to be specified that go beyond the simple patterns learnable by HMMs. Researchers have discovered that it is possible to combine the ME and HMM frameworks to create a more versatile framework known as a

Maximum Entropy Hidden Markov Model (MEMM) (Sangwan & Hansen, 2009). A basic conceptual depiction of an MEMM can be found in Figure 26 below.

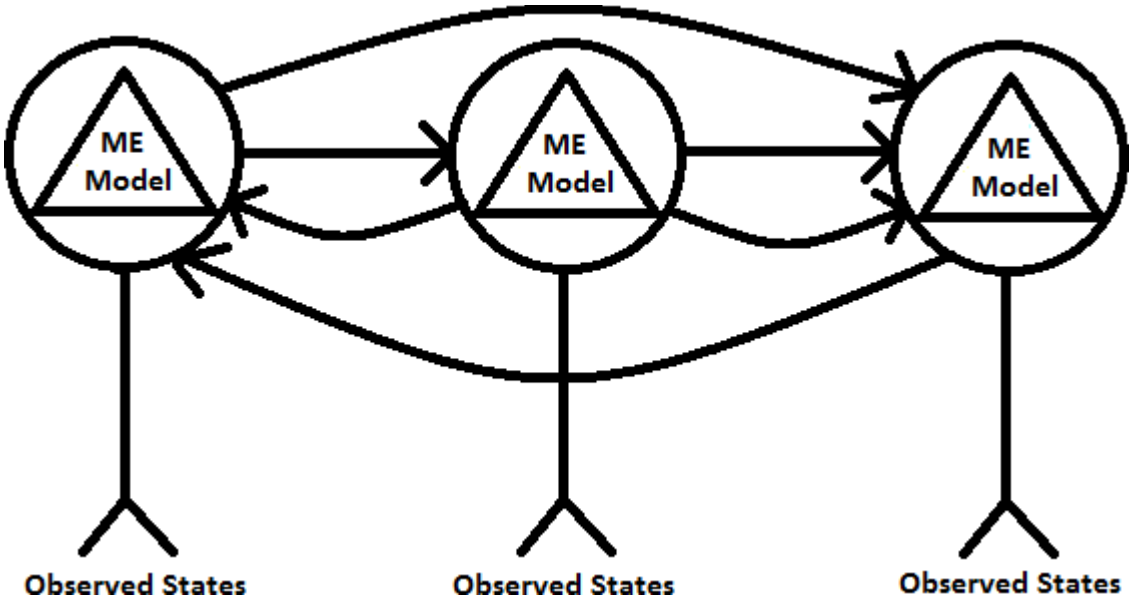

Figure 26. A depiction of a generic MEMM. The circles represent HMM hidden states.

MEMMs are similar in structure to a standard HMM, but each hidden state in the HMM is an ME classifier. This arrangement has the effect of enabling each classifier to condition the classifications of the subsequent classifier. In this way, the ME framework can be extended to sequential classification tasks and other dynamic modeling applications. There are tractable learning algorithms for this type of model, including variations that involve longer-distance dependencies between model states (Rosenberg, Klein, & Taskar, 2007), which allow it to be purposed for many of the same tasks that pure ME models or HMMs can perform. As an example, MEMMs have been used to perform PoS tagging (Toutanova & Manning, 2000). The accuracy of MEMM PoS taggers, 96.83%, is slightly less than the performance of HMMs. In addition, MEMMs have been deployed to perform information extraction and text segmentation (McCallum, Freitag, & Pereira, 2000) and semantic role labeling (Blunsom, 2004).

It is also possible to invert the way that ME and HMM models are combined by arranging for HMMs to act as the feature components of an ME model (Kazama, Miyao, & Tsujii, 2001). This method has been employed to perform word segmentation, a task

on which it achieved 95.18% accuracy. The purported benefit of this technique is that it allows for high accuracy even when the training corpus size is small. Clearly, this would be of benefit to the NLP field, as the creation of appropriate corpora can be expensive.

Yet another configuration is to use an ME framework to re-rank the outputs of a standard HMM (Highfill, 2011). For instance, if the n-best results are returned from applying the Viterbi algorithm to the HMM, then the ME model might decide that one of those results, not necessarily the best from the perspective of the Viterbi algorithm, is superior to the others. This type of configuration is capable of 91.8% accuracy on the PoS tagging task, which is somewhat below the other methods described.

Alternatively, researchers have developed techniques to learn HMM parameters directly using Maximum Entropy-inspired approaches (Walder, Kootsookos, & Lovell, 2003). This style of HMM can aid in the construction of models that are capable of avoiding overfitting and so then can be applied to more general types of tasks. Entropic training regimes were shown to outperform the Baum-Welch HMM algorithm under certain circumstances. However, the exact calculations of entropy measures that are required for the algorithm have proven to be intractable and are left as a topic for future work.

11.3 Criticisms of the ME Framework

Even though ME models (in particular, MEMMs) are generally able to incorporate a wider variety of information about the data they are modeling than their HMM counterparts, severe limitations remain. ME-based models are only able to account for hypothesis spaces that consist entirely of predefined features. If a modeler neglects to specify an important feature for classifying the data, then the resulting model will have suboptimal performance. Moreover, feature templates, while useful for defining large numbers of features, do not directly solve this problem, because only features that can be derived from those templates can be used. Unlike humans (Ormrod & Davis, 2004), ME models are not able to self-extend by learning new types of constraints beyond those that were specified in their initial designs. Instead, like all knowledge-lean approaches to NLP, any modification of models must be done manually by humans. Even though ME models are knowledge-lean themselves, it often falls upon

human researchers to augment them in an effort to improve performance (Toutanova & Manning, 2000).

While the ME framework is very efficient, it is still subject to restrictions on the number of features that can be feasibly entertained for a given model. It is not possible to include all possible features that might contribute to the classification of data; in PoS tagging, for instance, an exhaustive feature space would include all possible combinations of words before and after each word in a corpus. Furthermore, while the features utilized by an ME model do surpass the rudimentary lexical occurrences of HMMs, this does not enable them to consider features that are not readily observable from the data. For instance, if word sense is to be used as a feature, then the training corpora must be tagged with that information or some separate utility must be available to provide that information. Since large-scale manual tagging of knowledge-rich features is not currently efficient enough to be practice, ME models are severely limited as to the sophistication of their constraints. Moreover, even seemingly innocuous features, such as "this sentence has a verb" are difficult to encode in ME models. It is possible that a re-ranking ME model could employ this kind of feature to promote certain parses, but these types of features are rarely taken into account.

An additional criticism of ME models from the perspective of cognitive plausibility is that ME learning algorithms, namely the iterative scaling method used to determine the weights of features, is an inherently "off-line" learning process (Jager, 2007). Due to the fact that of iterative scaling methods require information about the features that cannot be known ahead of processing a training corpus, it is necessary to obtain this information by accessing and computing over the entire training corpus to establish feature parameters before learning can occur with those features. Even less cognitive plausible is the principle of maximum entropy itself. Humans are known for skewing distributions of data during learning an inference. This phenomenon occurs in a number of biases: the availability bias (Tversky & Kahneman, 1973), the hindsight bias (Hertwig, Fanselow, & Hoffrage, 2003), and the sunk cost fallacy (Arkes & Ayton, 1999), among others.

Furthermore, when humans are exposed to unknown or previously un-encountered conditions, they often impose a large amount of structure that is based on

analogical mappings to other experiences (Holyoak, Gentner, & Kokinov, 2001). Humans also will rigorously experiment with different outcomes and simulations when encountered with a novel situation in an attempt to discover new principles (Carbonell & Gil, 1990). A discussion of how well ME models can capture the structure of natural languages is further discussed in (Jager, 2007). From the analysis in that work, ME models are revealed to likely not be the best implementation of Optimality theory for modeling human languages. Rather, Stochastic Optimality Theory is proposed as a more faithful model of human languages; ME models are usually selected above other alternatives because they have a well-characterized learning algorithm. Finally, ME models fundamentally differ from humans because of their reliance on a perfect representation of a training corpus to successively augment themselves with new (Della Pietra, Della Pietra, & Lafferty, 1997). This practice is fundamental for avoiding the overfitting problem when using ME models. Feature selection assumes that the gain of each feature can be evaluated in isolation from the other features—this is almost certainly not true for anything beyond rudimentary models.

In terms of machine learning theory, there are some additional points that can be made against the ME framework. Despite the goal to counteract some of the drawbacks of the HMM framework, it has been discovered that ME models are reducible to HMM-like models (Goodman, 2002). Thus, it would not be expected that ME models are capable of fundamentally outclassing pure HMM-type models, especially given the huge number of variations of HMM models that are described above. Like most formalisms use in knowledge-lean NLP and AI, the best that can be said for ME models is that they address a few specific shortcomings that plague other approaches and provide these solutions in an efficient way. For example, an ME model that is converted into an HMM-like model might become totally intractable after the conversion due to state space explosion. Because ME models are discriminative, they must be trained using supervised learning techniques. Although MEMMs can learn without supervision (due to the incorporation of Markov structure and the Baum-Welch algorithm), there are few examples of this in practice (Peng, 2001), thus making it difficult to directly assess the utility of such methods.

Finally, there are purely mathematical reasons to avoid the use of ME models for modeling information-based data (Shalizi, 2011). The use of ME models inevitably results in assuming that the distribution of data being learning conforms to the family of exponential distributions. This apparently works well for statistical mechanics, but not for statistical inference. For instance, many types of phenomena are not well-characterized by exponential distributions. Furthermore, it is known that exponential distributions often exhibit more than one mode value, which can cause an extreme amount of variance between the expected value extracted from the data and the most common values in the distribution itself. This situation can lead to the creation of models that are not faithful at all to the underlying distribution of the training data.

11.4 Overview of the ME Framework: A Case Study

From one perspective, the ME Modeling framework can be seen as a response to some of the major shortcomings of Markov Models. Indeed, ME Models are discriminative models, and as such, are capable of more tractably representing certain kinds of complex constraints than Markov Models can. This is a major advantage, which allows ME Models to, in theory, represent many different kinds of patterns that would be extremely helpful as incremental steps towards general purpose human-level NLP. And yet, recall that a similar claim can be made about Markov Models—the catch being that Markov Models experience a state space explosion whenever they are pushed too far beyond their original design parameters. ME Models share a similar characteristic, as will be demonstrated below.

Consider the following fact, which is true of the English language: some sentences do not contain a main verb. For instance, this can occur due to ellipsis between segments of dialogue within discourse: (Speaker 1): "*Which one do you want?*" (Speaker 2): "***[I want]*** *That one there.*" In this case, *I want* is elided from Speaker 2's response. This kind of verb-less sentence, however, is relatively rare; most English sentences (especially in formal texts) would be expected to have a main verb, unless if there was some kind of grammatical error. Therefore, there are a number of tasks that might rely on the conditional probability of whether or not a main verb appears in a sentence.

For instance, one task that might be helpful would be to train an ME Model to recognize if a sentence that was missing a verb was an incorrigible error or if the sentence was merely exhibiting ellipsis. There is no straightforward way to encode this into an ME Model, though, because of two factors. First, it is difficult or impossible for data-driven approaches to learn information that cannot be directly read off of the data source, which clearly is the case for a missing verb. Second, a large amount of world knowledge would have to be incorporated into such a model in order to detect whether or not a given instance of ellipsis is permissible. While one might first think that ellipsis detection can be distilled down to check if the sentence is part of a dialogue. This problem is even more difficult than it initially appears, since it is possible to have ellipsis outside of a dialogue and there may be ellipsis errors made in the transcription of a dialogue. In order to determine if an ellipsis is licensed in a given situation, a large amount of world knowledge is necessary.

In the example given, a ME Model would have to understand that it would be possible to "recover" the elided material in the second sentence by importing the verb *want* from the previous sentence, which is a question. Although ME Models are capable of incorporating more complex constraints than Markov Models, ME Models still typically are restricted to using constraints that involve structures that are explicitly encoded into a training corpus,[53] such as word endings or PoS tags. Clearly, elided material is not present in the corpus at all and the task of creating enough hand-annotated data to train any model on the world-knowledge involved would be prohibitively expensive and time-consuming. Even if this knowledge could be encoded, the resulting number of constraints would be staggering. One additional concern is that, in an effort to keep state-spaces small, most modeling frameworks, including ME Modeling, are geared towards only considering material in one sentence at a time. Therefore, it is unlikely that an ME Model would be able to perform this task anywhere near the level of humans.

Beyond the concerns of ellipsis and world-knowledge, it is also instructive to consider a slightly more prosaic example that is motivated by PoS tagging. ME Models are often used in PoS tagging tasks to a good level of success. Figure 27 depicts a possible ME Model for PoS tagging currently considering the sentence *The man walked*

[53] Recall that discriminative models generally require a high degree of supervision during learning.

to the store. Some example constraints for this model are depicted. When performing PoS tagging, it would be useful to incorporate some kind of belief revision capability into models so that subsequent tagging decisions are able to impact previous taggings. To see why this would be useful, consider again the case of the presence or absence of a verb in a sentence. Since almost all commonly encountered sentences contain a verb, it seems as though the probability of a given tagging should be low if it does not tag at least some word as a verb. If the tagger reaches the end of a sentence without identifying a verb there are two possibilities: 1.) the tagger made a mistake or 2.) the sentence did not contain a verb due to an error or ellipsis.

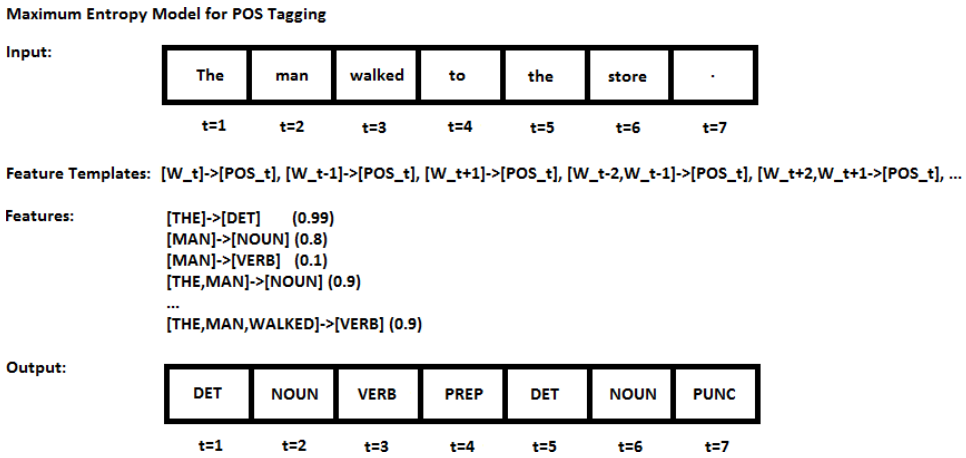

Figure 27. A Maximum Entropy Model applied to Part of Speech tagging.

As with many NLP problems, it is not useful to hardcode a single solution for all cases—here, it is undesirable to expect that all sentences must always have a verb. Yet, it also is sensible to have some provision for handling this kind of unlikely case. The question then is whether or not the ME Modeling framework can efficiently handle this kind of constraint. Recall again that in examining the capabilities of knowledge-lean methods, the principal question is not if a given approach is theoretically able to include a given constraint, but rather if a given approach can practically include a given constraint. At the very least, it is necessary to create constraints that somehow link the length of a sentence to the probability that a verb has occurred by some position within that sentence. Even for Maximum Entropy Models, this results in an intractable number of constraints because: 1.) the size of sentences is essentially unbounded and 2.) many of the constraint

features would have to be populated. Remember that the chief benefit of using a discriminative modeling framework (such as ME Modeling) is that a large number of constraint features can be defined without causing a state space explosion since only the features and constraints that are actually encountered in the data need to be instantiated in the final model; generative frameworks must instantiate all defined constraints, even if some of them never occur during training. But, the demand of representing every possible length of sentence is taxing even for a discriminative modeling framework. An illustration of this kind of state space explosion can be found in Figure 28. The constraints in red are those that deal with sentence length.

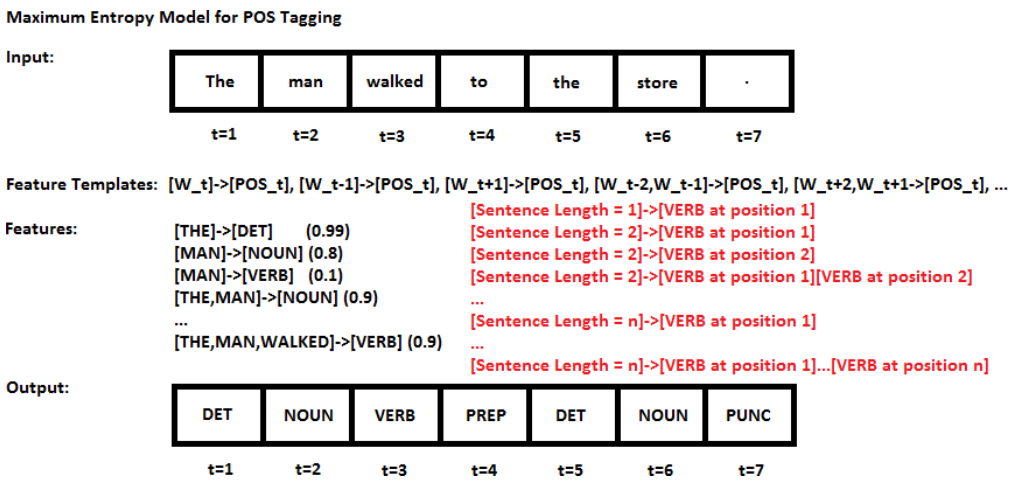

Figure 28. An example of how additional knowledge can explode a Maximum Entropy Model.

Ultimately, this case study has demonstrated that while ME Models are superior in some ways to Markov Models, they are far from being able to solve many of the more complicated problems that occur in human-level NLP. Other approaches attempt to deal with some of the issues of ME Models, but, as will be seen below, they all miss the mark for similar reasons. The most common of these reasons is the failure to accommodate certain kinds of information (e.g. sentence length or world-knowledge) without creating a state space explosion.

12. Conditional Random Fields (CRF)

Conditional Random Fields (Lafferty, McCallum, & Pereira, 2001) were developed to address some of the more qualitative shortcomings of HMMs. Namely, HMMs (including MEMMs) are subject to a quirk in inference known as the label bias. Inference in probabilistic graphical models can be conceptualized as a flow of probability mass through a network. In some cases, events with a low probability will be ignored during inference because the structure of HMMs normalizes the probability at each step in the sequence. On the other hand, CRFs are designed to normalize in a global sense, thus overcoming the label bias. The avoidance of the label bias allows CRFs to globally optimize for the best path during the inference process. However, CRFs are limited by the fact that they have an inherently high computational complexity, which increases the amount of time required to train such models (La, Guo, & Cao, 2012). CRFs have been used in a large variety of NLP tasks, such as determining the temporal structure of events (Kolya, Ekbal, & Bandyopadhyay, 2010) and shallow parsing (Sha & Pereira, 2003). A basic depiction of a CRF can be found in Figure 29.

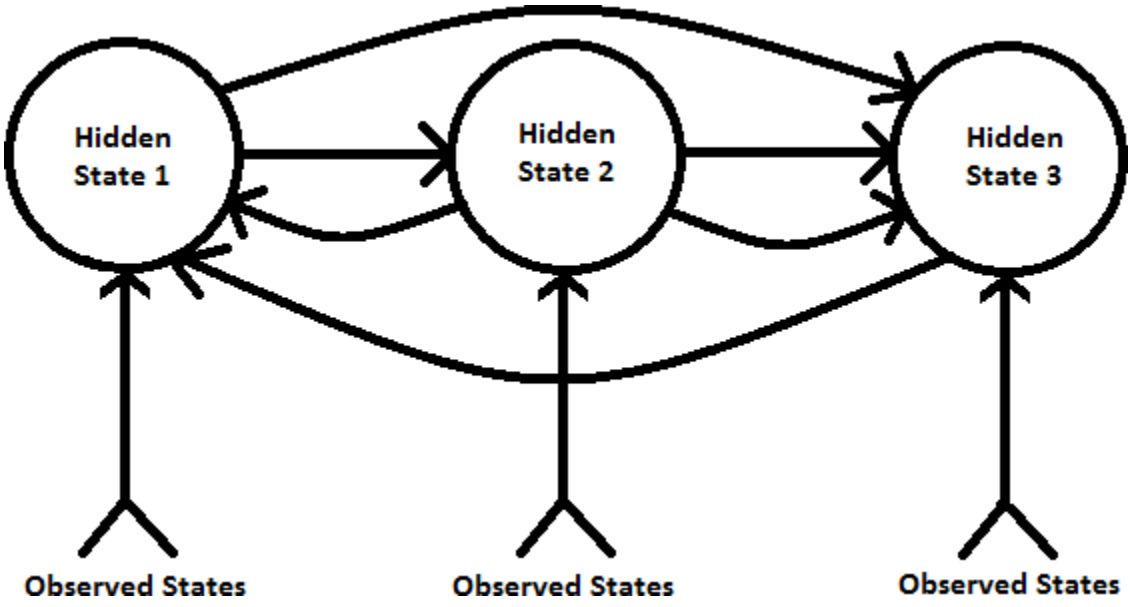

Figure 29. An illustration of a simple Conditional Random Field Model. Note the similarities to HMMs. The key difference between HMMs and Conditional Random Field Models is that the direction of influence between hidden states and observed states is reversed.

There have been some cross-model comparisons of HMMs, ME models and CRFs to determine the relative strengths and weaknesses of these formalisms. One such study evaluates how these three methods perform on detecting disfluencies[54] in speech (Liu, Shriberg, Stolcke, & Harper, 2005). That experiment reports results by error rate, not accuracy. Thus, by examining the results, it is clear that CRFs are capable of better performance than HMMs or ME models on 3 out of the four tasks. The researchers, however, cite the long training times for CRF models as a major drawback. Additionally, there is no guarantee that using a complex, specially-tailored model formalism will lead to performance improvements. For instance, this paper (Truyen, Phung, Bui, & Venkatesh, 2009), which applies a variation of CRFs to a noun phrase chunking task, does not achieve state-of-the-art performance, according to ACLWiki. Also, no explanation appears to be provided for why the experimenters capped their training corpora size at the order of 1000 sentences—this leads to doubts about the tractability of their method.

[54] E.g. *um*, *uh*, etc.

13. Markov Random Fields and Directed Markov Random Fields

Even though MEMMs are used in a number of state-of-the-art NLP systems, it has been discovered that there are some inherent negative consequences to the design of these models. In order to understand these flaws and the alternative methods that have been proposed to address them, some knowledge of the mathematics of random fields is required. A random field is a broader extension to the notion of a stochastic process—stochastic processes are structured by time, whereas random fields can be structured by more arbitrary spaces (Adler & Taylor, 2007). The main difference between an HMM and MRF and CRF-based models lies in the directedness of probabilities between hidden states and observed states. In HMMs, hidden states directionally condition observed states, and so require the modeling of the total join distribution between hidden and observed states. CRF models are arranged so that observed global states condition hidden states; in this way, CRFs can be said to be discriminative (Gong & Xiang, 2011). MRF fields have undirected dependency links, which make them more general than either HMMs or CRFs (Kindermann & Snell, 1980). A depiction of a simple MRF model can be found in Figure 30.

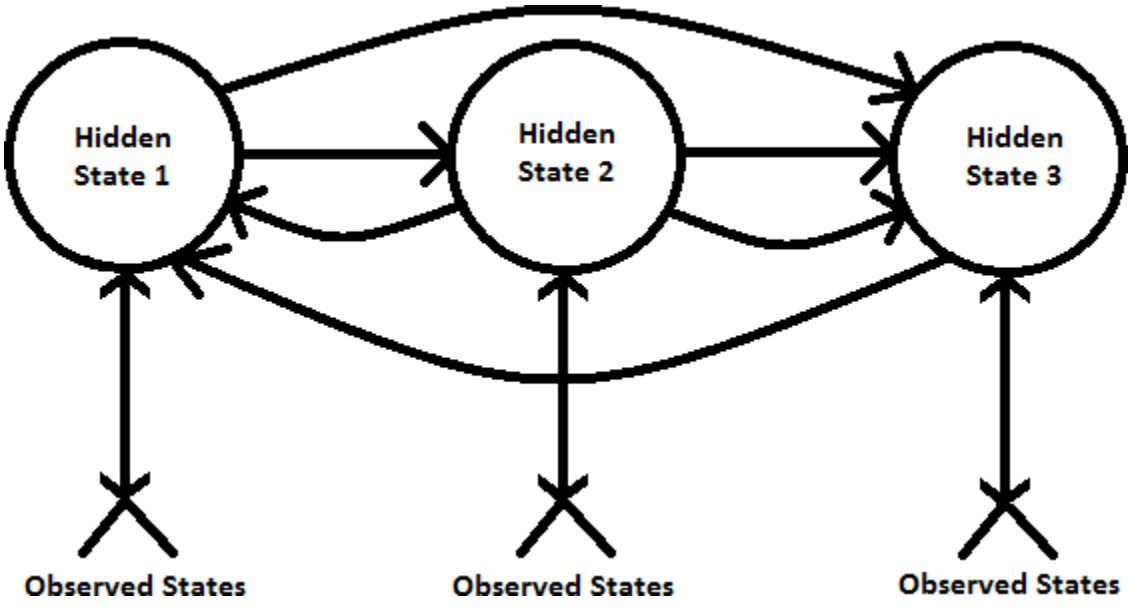

Figure 30. An illustration of a simple Markov Random Field Model. Note the similarities with CRFs and HMMs. In this case, the influence between hidden and observed states is bidirectional.

However, the generality of MRFs enables them to be hybridized with other types of knowledge representation and reasoning formalisms. One of the most prominent of these hybrids is Markov Logic, which has been extensively developed in an ongoing project called Alchemy (Alchemy Research Group) (c.f. (Richardson & Domingos, 2006)). While Alchemy has been employed for NLP problem solving (Singla & Domingos, 2006) (Poon & Domingos, 2009), it has a number of severe limitations that arise from its complexity. Like all other types of knowledge-lean models, it is unable to meaningfully extend itself with novel constraints. Furthermore, the complexity of inference in the Alchemy system requires it to rely on Monte Carlo methods, such as WalkSAT (Selman, Kautz, & Cohen, 1993). The use of WalkSAT requires modelers to specify an a priori threshold model cost (in terms of the total weight of broken constraints) so that the algorithm is capable of stopping (it is rare to have a complex model instance not break any of its constraints). Specifying such a threshold is difficult, if not impossible for NLP problems.

Furthermore, WalkSAT is not a systematic solver and therefore cannot exploit structure in inference processes that inevitably arise in language. An example of how systematicity is used in other parsing techniques can be found in papers on LR-parsing (Knuth, 1965) and chart-parsing (Charniak, Goldwater, & Johnson, 1998). Alchemy also is unable to perform certain types of inferences altogether.

For instance, Alchemy is incapable of properly representing object identity. If there are two broken constraints in an Alchemy model that differ only by one object, then the weights of those constraints should intuitively be counted only once instead of twice. This can have a tremendous impact on the results of a model. Consider the following scenario. An agent is playing Russian roulette with a six-shooter gun that is being fired in another room. The agent hears two shots and originally believes that it has a 1/4 probability of being shot if it pulls the trigger. However, the agent later is informed that the shot had an echo that it confused as a separate shot. The agent should therefore revise the probability of being shot to 1/5. This ability to revise is predicated on the ability to properly account for the identity of objects during inference. Finally, Alchemy models must be fully grounded in order for inference to occur. Attempting to eagerly ground

seemingly innocuous constraints can lead to a catastrophic explosion in model size. Consider the following ungrounded constraint:

$$Location(?x1, ?p1, ?t1) \wedge Location(?x2, ?p2, ?t2) \wedge \neg Same(?p1, ?p2, E)$$
$$\wedge \neg Same(?x1, ?x2, E) \overset{(\infty)}{\Longrightarrow} \neg Same(?t1, ?t2, E)$$

Figure 31. An ungrounded constraint with object identity considerations.

This constraint states the commonsense rule that if there are two objects, x1 and x2, that are known to be at two different locations, p1 and p2, and x1 and x2 are the same object, then the objects cannot be at those locations at times t1 and t2 unless if t1 and t2 are not the same time[55]. If there are 10 objects, 100 locations, and 100 times, the total number of grounded constraints would be: 10 * 100 * 100 * 10 * 100 * 100 = 10, 000, 000, 000 constraints. The need to ground all constraints ahead of inference also prevents Markov Logic models from using crucial kinds of representations that can greatly improve the efficiency of problem solving (Bignoli, Cassimatis, & Murugesan, 2010).

[55] I.e. Something cannot be in two places at once.

14. Probabilistic Context-Free Grammars (PCFG)

Probabilistic Context-Free Grammars (also known as Stochastic Context-Free Grammars) are mathematical structures that correspond to Deterministic Context-Free Grammars, except that each phrase production in a PCFG is associated with a probability[56]. Recall that CFGs describe languages that belong to the family of complexities that can be described by a set of phrase production rules that are subject to certain limitations. Namely, the production rules must have a single non-terminal symbol on the left-hand side and a sequence of terminal and non-terminal symbols on the right-hand side. Terminal symbols correspond to actual words in the language—in natural languages, terminal symbols correspond to lexical items. Non-terminal symbols correspond to abstract structures that impose boundaries on groups of terminal and non-terminals—in natural languages, non-terminal symbols correspond to phrases and sentences (the sentence-level is typically understood to be the unique start symbol for a natural language CFG). Sentences in CFGs can be represented as a parse tree. For example, the sentence *I saw the man with a telescope* could be represented by the following parse tree:

[56] Alternatively, Deterministic CFGs can be considered to be PCFGs in which all productions are: 1.) non-conflicting and 2.) associated with a probability of 1.0.

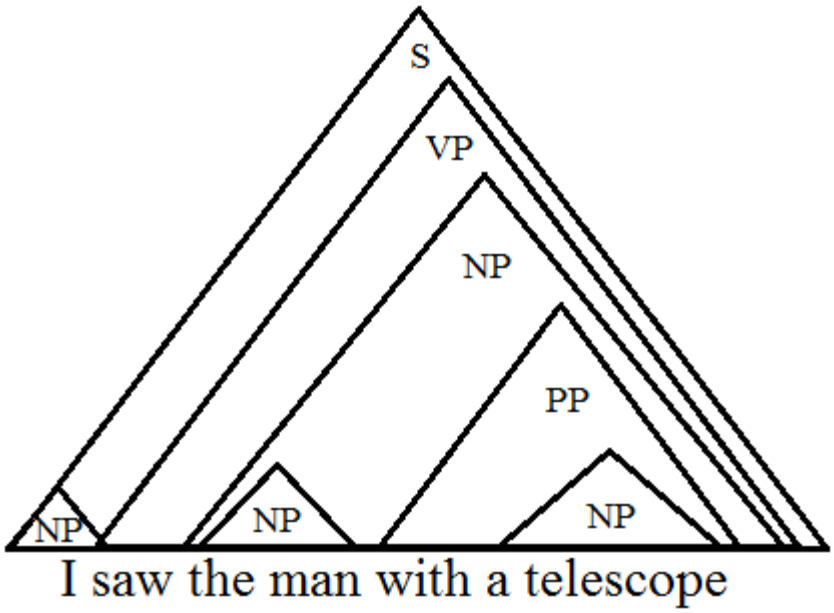

Figure 30. A graphical representation of "I saw the man with a telescope" broken down into phrase structures.

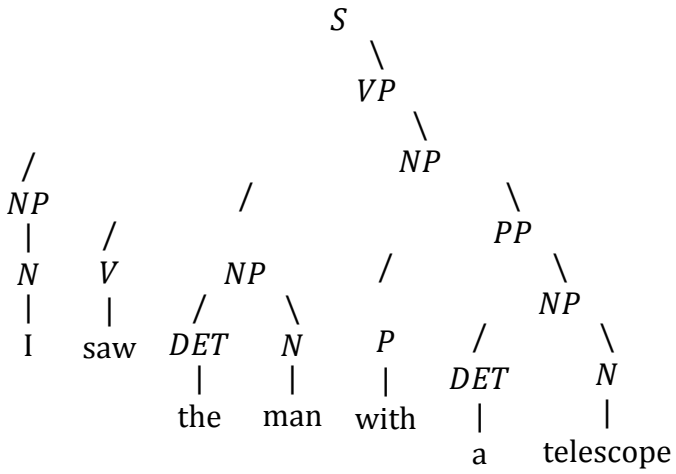

Figure 31. A CFG-style parse tree for "I saw the man with a telescope."

The terminal symbols are contained in the terminal nodes of the parse tree; non-terminal symbols are contained in non-terminal nodes. If a language consisted precisely of the sentence above, then the production rules would be:

S	$(1.0) \rightarrow$	$NP\ VP$
VP	$(1.0) \rightarrow$	$V\ NP$
NP	$(0.25) \rightarrow$	N
NP	$(0.25) \rightarrow$	$NP\ PP$
NP	$(0.50) \rightarrow$	$DET\ N$
PP	$(1.0) \rightarrow$	$P\ NP$
V	$(1.0) \rightarrow$	saw
N	$(0.33) \rightarrow$	I
N	$(0.33) \rightarrow$	man
N	$(0.33) \rightarrow$	telescope
P	$(1.0) \rightarrow$	with
DET	$(0.5) \rightarrow$	the
DET	$(0.5) \rightarrow$	a

Figure 32. Productions rules for the one sentence language.

Note that all of the productions' probabilities can be calculated by dividing each unique left-hand symbol with the possible right-hand sequences that follow from the left-hand symbol. There are no possible ambiguities or decision points in a language with one sentence and one parse. There is also no possibility that there are unseen elements of the grammar. To show a trivial example of how adding new sentences would alter the production rules, consider the sentence *I saw the man with a hat*. The parse tree for this sentence would be:

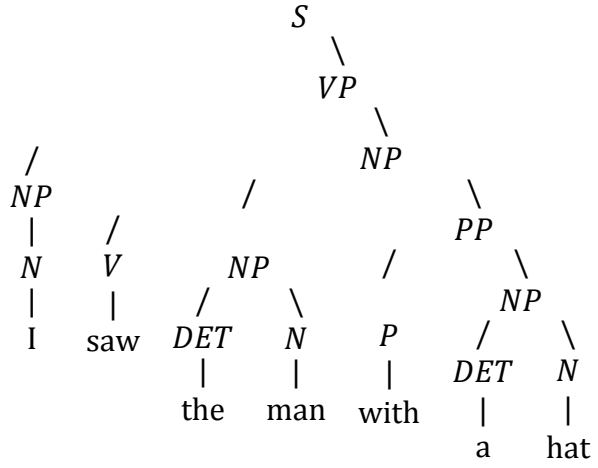

Figure 33. A CFG-style parse tree for "I saw the man with a hat."

Clearly, this is almost identical to the previous parse tree. Because the only difference between this sentence and the sentence above is the final terminal, the production rules that capture the expanded language do not change except for a new rule from *N* and new probabilities for the *N* productions:

$$
\begin{array}{lll}
S & (1.0) \rightarrow & NP\ VP \\
VP & (1.0) \rightarrow & V\ NP \\
NP & (0.25) \rightarrow & N \\
NP & (0.25) \rightarrow & NP\ PP \\
NP & (0.50) \rightarrow & DET\ N \\
PP & (1.0) \rightarrow & P\ NP \\
V & (1.0) \rightarrow & \text{saw} \\
N & (0.33) \rightarrow & \text{I} \\
N & (0.33) \rightarrow & \text{man} \\
N & (0.16) \rightarrow & \text{telescope} \\
N & (0.16) \rightarrow & \text{hat} \\
P & (1.0) \rightarrow & \text{with} \\
DET & (0.5) \rightarrow & \text{the} \\
DET & (0.5) \rightarrow & \text{a}
\end{array}
$$

Figure 34. Productions rules for the two sentence language.

Now consider if an alternative parse of the first sentence, *I saw the man with a telescope*, was possible:

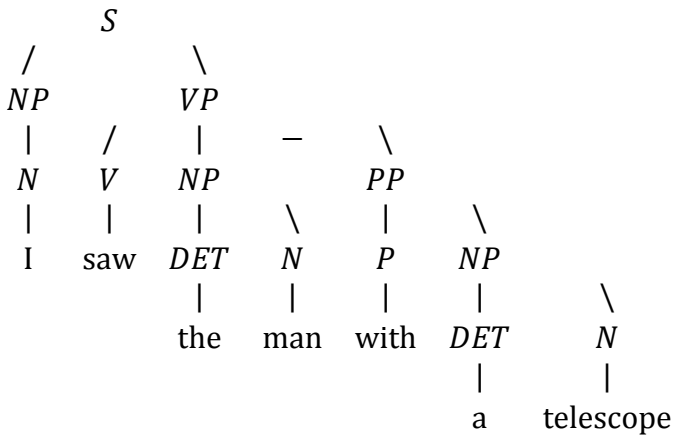

Figure 35. A CFG-style parse tree for an alternative parse of "I saw the man with a telescope."

Although the terminals of this new sentence are identical to the terminals of the first sentence, the non-terminal structure is different. Conceptually, the first sentence attaches the prepositional phrase *with a telescope* to the noun phrase *the man* to create a larger noun phrase *the man with a telescope*. Thus, what is seen is a man and an accompanying telescope. The third sentence, however, attaches the *with a telescope* phrase directly to the verb *saw* to create a more complex verb phrase. In this case, the prepositional phrase modifies the verb to indicate how the man was seen—by the use of a telescope by the subject of the sentence. The changes to the productions rules in this case are captured as follows:

$$
\begin{array}{ccc}
S & (1.0) \rightarrow & NP\ VP \\
VP & (0.66) \rightarrow & V\ NP \\
VP & (0.33) \rightarrow & V\ NP\ PP \\
NP & (0.27) \rightarrow & N \\
NP & (0.55) \rightarrow & NP\ PP \\
NP & (0.18) \rightarrow & DET\ N \\
PP & (1.0) \rightarrow & P\ NP \\
V & (1.0) \rightarrow & \text{saw} \\
N & (0.33) \rightarrow & \text{I} \\
N & (0.33) \rightarrow & \text{man} \\
N & (0.22) \rightarrow & \text{telescope} \\
N & (0.11) \rightarrow & \text{hat} \\
P & (1.0) \rightarrow & \text{with} \\
DET & (0.5) \rightarrow & \text{the} \\
DET & (0.5) \rightarrow & \text{a}
\end{array}
$$

Figure 36. Production rules for the three sentence language.

There are several methods that can be used to deduce the structure of a PCFG from a collection of sentences. By far the most common method is to train a model using a Treebank with parsing information supplied as bracketing annotation. In the case of the Penn Treebank, this setup is very similar to the data used to train PoS taggers, but a new kind of marking protocol is used (Bies, Ferguson, Katz, & MacIntyre, 1995). For

instance, the sentence *I saw the man with a telescope* may be bracketed in either of the following ways[57]:

$$VP\ Attachment\text{: } \big(VP \text{ saw } (NP \text{ the man})\big(PP \text{ with } (NP \text{ a telescope})\big)\big)$$

$$NP\ Attachment\text{: } \big(VP \text{ saw } (NP\ (N \text{ the man})\big(PP \text{ with } (NP \text{ a telescope})\big)\big)$$

Figure 37. Possible phrase bracketing solutions for "saw the man with a telescope."

To learn this structure from marked text, it is possible to use a modified version of the parser that operates over deterministic CFGs. This procedure is known as the CYK algorithm (Kasami, 1965) and can be run in cubic time. An example of how this basic kind of parser can be applied to process PCFGs is provided in: (Chappelier & Rajman, 1998). State-of-the-art parsers no longer use plain CFG parsing, but a further examination of why this is the case is warranted.

The limitations of PCFGs are well-known and are even routinely covered in introductory courses for statistical NLP (Dickinson, 2006).

There are three main points that decrease the usefulness of PCFGs for modeling natural languages. First, the class of CFGs is limited in the sense that they have no way to represent that one subtree can condition another subtree. In the sentences above, this means that it is impossible to utilize the probability that a verb phrase follows an indefinite noun phrase. Second, PCFGs are inherently unlexicalized, which prevents production rules from taking into account information about how the presence of a terminal symbol (i.e. lexeme) under the scope of a non-terminal symbol impacts the probability of productions involving that non-terminal symbol. Thus, there is no way to indicate that the verb *see* is a transitive verb and would therefore be likely to take another argument besides the subject. Out of all of the shortcomings of PCFGs, the problem of lexicalization is probably the best addressed in the literature. A section below will be devoted to the discussion of lexicalized PCFGs. Third, world knowledge and semantic relationships are not considered by standard PCFGs. Such information is crucial for disambiguating alternative parses of natural language sentences. For example, world

[57] Adapted from (Manning & Schütze, Foundations of statistical natural language processing, 1999)

knowledge informs a human that the prepositional phrase *with a hat* is unlikely to directly modify the verb *see* because hats are not instruments that can be used to see things. This final shortcoming is the most serious flaw and the one that cannot truly be addressed by any dedicatedly knowledge-lean approach.

Unfortunately, these shortcomings are the very properties that allow PCFGs to be computationally tractable to learn and to perform inference with. Chomsky made some initial attempts to extend the applicability of CFGs to NLP by such techniques as adding transformational rules to grammars (Chomsky, 2002). However, there is evidence that natural languages lie completely outside the class of CFGs and such measures are ultimately futile in the quest for human-level performance.

Even worse than the inherent limitations of the PCFG casting of natural language parsing is that the metrics used to gauge the fitness of parses are just as misleading as those used to measure the effectiveness of PoS taggers. Although this issue was covered extensively above, it is worthwhile to revisit it briefly here. Consider the effect of using the PARSEVAL metric, which is the most widely used, to determine the accuracy of a parser for the toy language described above. Because of the way that PARSEVAL evaluates parses (i.e. by checking phrase boundaries and favoring local structures over global completeness), it often reports greatly inflated scores compared to what an impartial observer would expect from the actual operation of the parser. As an example, one could create a simple parser for the three-sentence toy language that was described above. This parser would always bind a PP to a preceding VP instead of a preceding NP if such an ambiguity is encountered. Under this scheme, two of the three sentences would be parsed incorrectly,[58] resulting in a per-sentence accuracy of 33.3%. However, PARSEVAL would rate this parser as having a 94.4% F-1 score accuracy. Although it is useful to know the PARSEVAL score, as it provides a finer-grained insight into the performance of a parser, it clearly is misleading to suppress the per-sentence accuracy, which may be radically different. Partially correct parses can still be useful for some applications, but in order to use them in any significant way, it is necessary to know something about how much of the intended meaning that the parse has preserved. The

[58] These sentences are I saw the man with the hat and I saw the man with the telescope, when with the telescope modifies the man.

ramifications of even small mistakes can be serious; for instance, the PP attachment in the sentence *Approach the man with the tank* is potentially very important.

These concerns raised above occur with real parsers, as well. According to the ACLwiki (ACLWiki, 2012), the state-of-the-art performance for syntactic parsers range from 90.1% to 92.1%. These approaches all use some version of PCFG parsing, but these levels of performance are obtained only by making significant modifications to the PCFG learning algorithms, some of which are described below. Therefore, it is entirely conceivable that these parsers are processing every single sentence incorrectly, although, in reality, this is not the case. A cursory examination of a parser that has a reported accuracy of 91.4% (Charniak & Johnson, 2005) reveals that the per-sentence accuracy is actually 62.3%[59]. Information about the parser's performance on individual sentences can be found at (Johnson M.).

14.1 Beyond PCFGs

The next level in complexity fully beyond CFGs in the Chomsky hierarchy is the family of Context-Sensitive Grammars (CFGs) (Rozenberg & Salomaa, 1997). One of the primary differences between CFGs and CSGs is that CSG production rules can contain multiple terminal and non-terminal symbols on the left-hand side of rules:

$$\text{an}NP \rightarrow \text{an aardvark}, \ldots, \text{an aztec}$$
$$\text{a}NP \rightarrow \text{a baal}, \ldots, \text{a bzou}$$

Figure 38. Examples of CSG production rules

Thus, the context of a non-terminal symbol can affect the types of production rules that follow from it. In the example above, NPs that are prefixed with the terminal symbol *an* lead to NPs with nouns that begin with 'a,' whereas NPs that are prefixed with the non-terminal symbol *a* lead to NPs with nouns that begin with 'b.' While natural language comprehension is clearly context-based to some extent, it is very difficult to use unmodified CSG parsers, because their string recognition procedure belong to the P-SPACE complete complexity class (Karp, 1972). There are no known efficient

[59] Out of 3712 parsed sentences, 2311 sentences match the gold standard parse.

algorithms for solving general P-SPACE complete problems; some P-SPACE problems can be posed in a way that there do not even exist efficient approximation algorithms to solve them (Marathe, Hunt III, & Ravi, 1993).

However, part of the high complexity of CSGs is due to the fact that they define languages that are not like any known natural language in their structure[60]. This suggests that there may be a class in between CFGs and CSGs that might represent many of the features of natural languages without having to consider the worst-case complexity languages in the CSG family (Kasai, 1970). Tree-adjoining grammars (Joshi & Schabes, 1997) and mildly context-sensitive grammars (Stabler, 2004) offer a more tractable alternative to full context-sensitive languages. These methods are able to provide advancements over other techniques that rely on a simpler construal of natural languages. For instance, nearly context-free grammars have been used on syntactic parsing tasks (Charniak, 2010). This method yields an accuracy of 89.4%, which is actually less than the modified PCFG results cited above.

Other efforts to improve on parsing with "pseudo-Context Sensitive Grammars" have proven somewhat successful in terms of boosting accuracy (Charniak & Carroll, 1994). However, the authors demonstrate that the rules induced by such parsers can be incorporated into a standard PCFG parser, so it is clear that there is not much of an advance in terms of actual complexity. Long sentences were also eliminated from this analysis, which makes it dubious that this approach will be able to solve all or even most of the short comings of standard PCFG parsers. On the other hand, information extraction with kernel-based methods (Zelenko, Aone, & Richardella, 2003) that utilize CSGs (Zhou, Zhang, Ji, & Zhu, 2007) performs at an accuracy level of 73.2% on a task that the previous state-of-the-art performed at a 63.6% accuracy level. Another study involving a data modeling formalism[61] of CSG information extraction reports that the model it describes is capable of explicitly accommodating several non-local features of

---

[60] The evolutionary view of language holds that features of natural languages are subject to fitness evaluations, just as standard biological traits are (Kirby, 2007). Thus, it is possible to conjecture that since extremely complex features of languages that serve no obvious communicative purpose in human-environments, especially ones that are not found in any extant language, are not necessary for human-level linguistic comprehension.

[61] As opposed to kernel methods, which are a type of algorithmic modeling (c.f. (Breiman, Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author), 2001)).

natural languages, such as long-distance subject-verb agreement. This represents a clear advance over standard PCFG parsing. Thus, it is apparent that near CSG-based approaches are superior for some NLP tasks. Nevertheless, there is a huge gap between human-level performance and this method, which already taxes the limits of modern computing technology. The authors themselves enumerate a list of linguistic phenomena, such as *wh*-form questions, which cannot yet be handled by their approach. Even if these more advanced parsing techniques could be extended to cover a large portion of natural language features, they would be subject to approximate algorithms for learning and inference that inevitably degrade performance (Hollingshead, Fisher, & Roark, 2005).

There are models for language that are beyond the complexity of CSGs. The most well-known example is HPSG (Sag, Wasow, & Bender, 2000), which, at least according to the most recent analysis of the decision problem for strings in those languages, has the same level of complexity as the language class that can be recognized by a Turing Machine (Hegner, 1997). While this is a very complete model of natural languages, it is currently impractical to implement a full HPSG parser, especially one that would rely on knowledge-lean techniques. Aside from the issues with computational intractability, there are no suitable large corpora that could support such a learning process, due to the complexity of the lexical entries and structure that would be required to train such a model. To account for the various types of interplay between syntax and semantics that are encountered in natural languages, it is almost certain to require some kind of knowledge-rich data. While this is not incompatible with the statistical approach to NLP in general, it definitely goes against the current trends in methodology in play in the field. Additional refinements to HPSG theory have recently been proposed by incorporating features of construction grammar theory with the base HPSG program (Boas & Sag, 2012). Again, while this certainly represents some kind of advance

14.2 Extensions to PCFGs

Even though PCFGs are incapable in themselves of capturing all of the nuances of human-languages, they can be extended to produce an approximate model of them. For this reason, they continue to be more popular than models for other levels of languages

with greater expressivity. The most common extensions to these are described in this following section.

Lexicalized PCFGs

Perhaps the most important advancement in PCFG parsing is the use of lexicalized production rules. A lexicalized production rule is similar to an unlexicalized one, except that each non-terminal symbol in a lexicalized rule is labeled with the terminal symbol (i.e. the lexical item) that is the "head" of the phrase that corresponds to the non-terminal symbol. This results in a much larger number of production rules, which helps to capture some of the finer structure in natural languages that arises because of the various semantic relationships between words and phrases. For example, a lexicalized version of the production rules for the toy language described in the section above would be:

$$
\begin{array}{ccc}
S_{\text{saw}} & (1.0) \rightarrow & NP_{\text{I}} VP_{\text{saw}} \\
VP_{\text{saw}} & (0.33) \rightarrow & V_{\text{saw}} NP_{\text{man}} \\
VP_{\text{saw}} & (0.33) \rightarrow & V_{\text{saw}} PP_{\text{hat}} \\
VP_{\text{saw}} & (0.33) \rightarrow & V_{\text{saw}} NP_{\text{man}} PP_{\text{telescope}} \\
NP_{\text{man}} & (1.0) \rightarrow & DET_{\text{the}} N_{\text{man}} \\
NP_{\text{telescope}} & (1.0) \rightarrow & DET_{\text{a}} N_{\text{telescope}} \\
NP_{\text{hat}} & (1.0) \rightarrow & DET_{\text{a}} N_{\text{hat}} \\
NP_{\text{I}} & (1.0) \rightarrow & NP_{\text{I}} \\
PP_{\text{telescope}} & (1.0) \rightarrow & P_{\text{with}} NP_{\text{telescope}} \\
PP_{\text{hat}} & (1.0) \rightarrow & P_{\text{with}} NP_{hat} \\
V_{\text{saw}} & (1.0) \rightarrow & \text{saw} \\
N_{\text{man}} & (1.0) \rightarrow & \text{man} \\
N_{\text{telescope}} & (1.0) \rightarrow & \text{telescope} \\
N_{\text{hat}} & (1.0) \rightarrow & \text{hat} \\
N_{\text{I}} & (1.0) \rightarrow & \text{I} \\
P_{\text{with}} & (1.0) \rightarrow & \text{with} \\
DET_{\text{the}} & (1.0) \rightarrow & \text{the} \\
DET_{\text{a}} & (1.0) \rightarrow & \text{a}
\end{array}
$$

Figure 39. Lexicalized production rules for the three sentence language.

The subscripts on the non-terminal symbols represent the lexical items. Since it is common practice to consider the main verb to be the head of a sentence and *saw* is the only main verb in the toy language, this explains why there are only sentence-level

production rules from an S_{saw} node to sequences of lexicalized NP and VP nodes. However, since NPs can be headed by *man*, *telescope*, or *hat*, there is more than one kind of production rules from a lexicalized NP symbol; recall that there was only one kind of NP in the unlexicalized version of the language. A unique start symbol is required for a CFG to be well-formed, so in general, a production from that symbol to all of the sentence-level symbols would be required. Because there is only one sentence-level symbol in the grammar, such a symbol is unnecessary.

Parse trees in a lexicalized CFG parse are similar to those that are produced by unlexicalized parses. The main different between them is that a lexicalized parse has phrase nodes that are subscripted with their head lexeme. For example, the above diagram shows one possible lexicalized parse for the sentence *I saw the man with a telescope*:

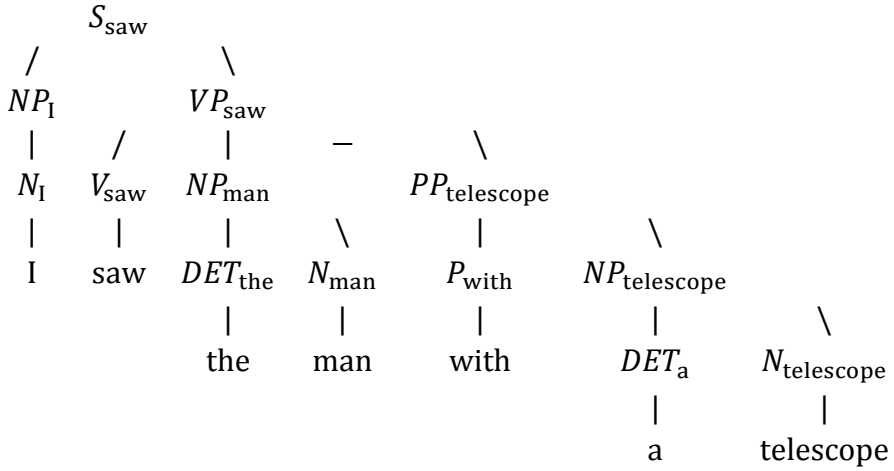

Figure 40. Lexicalized parse tree for "I saw the man with a telescope."

One of the best performing lexicalized PCFGs without other modifications is described in: (Collins, 1999). This parser achieves a labeled per-bracket accuracy of 84.7%[62]. Note that the labeled sentence accuracy is only 56.6%. It is worth noting that a more advanced unlexicalized parser that was developed more recently by the Stanford NLP group (Klein & Manning, 2003) is capable of exceeding this performance with an accuracy of 86.36%. Even though, intuitively, lexicalized information should make for a

[62] Calculated F1-score from a Labeled Recall of: 84.6% and a Labeled Precision of: 84.9%

vastly more powerful parser, this does not turn out to be the case. First, as was discussed in the section above on the PARSEVAL metric, the way that PCFG parsers are evaluated focuses heavily on phrase boundaries, which can be learned fairly well with or without lexicalization. Second, the large number of production rules in lexicalized grammars causes there to be a relatively small amount of information to set the probability for each rule. This second issue is a serious drawback, since it is endemic to PCFG parsing in general.

Many of the possible combinations of lexicalized phrases will not be encountered in any given corpus, so smoothing techniques must be used to estimate probabilities for these cases (c.f. (Jelinek, Merialdo, Roukos, & Strauss, 1990), which is still relevant despite the fact that the paper is about speech recognition). This type of estimation does not take any semantic information about lexical items into account, thereby greatly reducing the effectiveness of the lexicalization technique. Reliance on naïve smoothing techniques can result in significantly degraded accuracy. For instance, if there is no training evidence to rely on, then all unobserved sequences will be given the same probability, despite if there are possible clues in context or analogical structure mapping that could inform the parser's decision. Consider if the evaluation corpus had the following sentence: *He put the ball in the basket*. If neither ball nor basket was present in the training corpus, the PP attachment in this sentence would have to be resolved by appealing to the overall relative numbers of NP-PP vs. VP-PP attachments in the training data. Therefore, if more NP-PP attachments occurred, then this sentence would be parsed incorrectly (i.e. as *He put an entity called "ball in the basket"*). However, seeing examples of other uses of the verb *put* would not be able to condition the PP attachment probability without special direct intervention from a human programmer.

Humans, on the other hand, can readily learn such patterns by forming a generalized category of phrases by analyzing the use of the most commonly used verbs in ditransitive constructions (Goldberg A. E., 1999). Since PCFG parsers are unable to learn rules in this way, it is implausible that they would be able to capture at least some of the common structures that appear in natural languages. Moreover, new observations after learning are not incorporated into standard PCFG parsers (although self-training parses may be able to modify themselves to some degree—this is discussed below),

which certainly contravenes human cognition. Furthermore, studies of human sentence processing have revealed that making predictions is a crucial part of how constructions in language are learned (Goldberg, Casenhiser, & Sethuraman, 2005). Even lexicalized PCFG parsers do not learn in this way. Thus, lexicalization alone does not provide a total solution to the problem of human-level NLP.

14.3 Re-Ranked PCFGs with Maximum Entropy Modeling

Although PCFG parsers that incorporate lexicalized production rules significantly outperform their unlexicalized counterparts, they still are far from human-level capability. Since it is relatively inexpensive to generate a large number of possible CFG-level parses for a sentence, some researchers have exploited this fact to produce two-stage parsers that achieve greater accuracy (Charniak & Johnson, 2005). The first stage of these approaches is to general all possible parses of a sentence, not only the most probable ones. These parses are then re-ranked by applying a Maximum Entropy-like model that promotes or demotes the likelihood of each potential parse according to a set of features. Note that these features are subject to the same kinds of limitations as the features that were described in the section devoted to Maximum Entropy Modeling as an extension to HMMs.

Primarily, these constraints are knowledge-lean and take little semantic information into account. Any rectification of this situation would essentially be a backdoor into knowledge-rich modeling in contradiction to the established tenets of the knowledge-lean approach. However, there are issues even with the way that re-ranking parsers operate even if the question of semantic information is removed from consideration. One of the major inherent downsides of the re-ranking approach is that there are influences between structures in natural language discourse that cross the boundaries of sentences. Therefore, to cover such cases, an exhaustive parser would have to parse the cliques of mutually influencing sentences that appear in a corpus in all possible combinations. For larger texts, this clearly would lead to a combinatorial explosion in the number of parses that would have to be generated and re-ranked. Additionally, it seems cognitively implausible that human brains would waste time exhaustively generating all possible (syntactic) interpretations of a sentence before analyzing the communication at the

semantic or pragmatic levels of understanding. Not only would that type of arrangement be very inefficient in time, but also it might consume too much cognitive activity to be performed in the on-line scenarios where the comprehension of language is the more vital for survival.

14.4 Self-Training PCFGs

In order to provide more training examples for a PCFG, some attempts have been made to create self-learning algorithms; for instance, McClosky, Charniak, and Johnson's project (McClosky, Charniak, & Johnson, 2006). A self-learning algorithm would not require annotated sentences beyond its initial training. Therefore, if a model is trained on a moderately-sized, hand marked corpus, then it could be deployed to learn on vast amounts of more unstructured text. For instance, McClosky, et al. used approximately 2 million raw sentences to obtain a parser with slightly greater accuracy than a non-self-trained version of the same algorithm (c.f. (Charniak & Johnson, 2005)). The self-trained parser achieved an accuracy of 92.1%, whereas the original parser achieved 91.4%. This is not a large absolute improvement, but it is great enough to be statistically significant.

While this kind of method alleviates some of the issues surrounding the production of amply large corpora for training NLP models, it does not address many of the critical problems inherent in the knowledge-lean approach. For instance, semantic information and commonsense knowledge are not represented in this kind of model and probably cannot even be coded into a small seed training corpus that would be extensible to learning such relations in raw text. There are also many instances of linguistic information that is passed implicitly in constructions, such as ellipsis or anaphora, which are inherently opaque to a corpus-based approach.

14.5 Overview of PCFGs: A Case Study

Continuing in the pattern of the treatments of Markov Models and ME Models, it is useful here to consider what happens when some of the shortcomings of PCFG Models are directly confronted with a concrete example. In this case, the greatest challenge again is how to incorporate world-knowledge into a knowledge-lean framework without incurring a resulting state space explosion. As was mentioned above, the PCFG

framework is not conducive at all to take knowledge-rich structures into account during learning or inference. Lexicalization is the only major way to incorporate any such information and it only does so obliquely and with relatively poor results.

Consider again the *I saw the man with a telescope* example. There is real ambiguity here, namely, with the way that the prepositional phrase *with a telescope* attaches to the rest of the sentence. Either *with a telescope* attaches to the verb *saw* or the noun *man*. Now, consider a similar sentence: *I saw the mouse with a telescope*. Under normal circumstances, world knowledge informs humans that mice do not usually own telescopes (at least in the real world). However, there is no way to encode this kind of fine-grained relational knowledge in a lexicalized PCFG. At the very least, encoding all of the ways that world-knowledge can impact parse tree structure would require an impossibly large corpus.

But, just to completely make point of a fundamental flaw in the PCFG framework as it is applied to human-level NLP problems, one can attempt to answer how exactly to capture information about telescopes in this methodology. In First Order Logic, it is possible to express a wide variety of information about telescopes richly and succinctly. Example statements can be found in Equation 11.

$$IsA(m, mouse) \text{ ^ } IsA(t, telescope) \text{ ^ } Preposition(p, \text{“}with\text{”}) \text{ ^ } Event(e, seeingEvent) \Rightarrow Relation(m, t, \text{“}near\text{”})$$

$$IsA(m, cat) \text{ ^ } IsA(t, telescope) \text{ ^ } Preposition(p, \text{“}with\text{”}) \text{ ^ } Event(e, seeingEvent) \Rightarrow Relation(m, t, \text{“}near\text{”})$$

…

$$IsA(h, human) \text{ ^ } IsAIsA(t, telescope) \text{ ^ } Preposition(p, \text{“}with\text{”}) \text{ ^ } Event(e, seeingEvent) \Rightarrow Relation(h, t, \text{“}using\text{”})$$

Figure 41. Statements about telescopes in First Order Logic.

However, to use this kind of knowledge in a PCFG, these statements need to be wrapped into lexicalized production rules. Samples of this kind of production rule can be found below in Figure 44.

$$S[see]\ (0.999) \rightarrow NP[I]VP[saw]NP[mouse]PP[telescope]$$

$$S[see]\ (0.001) \rightarrow\ NP[I]VP[saw]NP[mouse]$$
$$S[see]\ (0.999) \rightarrow\ NP[I]VP[saw]NP[cat]PP[telescope]$$
$$S[see]\ (0.001) \rightarrow\ NP[I]VP[saw]NP[cat]$$
$$\ldots$$
$$S[see]\ (0.5) \rightarrow\ NP[I]VP[saw]NP[man]PP[telescope]$$
$$S[see]\ (0.5) \rightarrow\ NP[I]VP[saw]NP[man]$$

Figure 42. PCFG production rules about telescopes.

Recall from the discussion of lexicalized PCFGs that adding lexicalization information alone explodes the number of constraints to the point that there is not enough training data in corpora to obtain actually observed probabilities for many of these constraints. Instead, estimations of such probabilities must be made, which ultimately reduce the potential accuracy of lexicalize PCFGs. World-knowledge-equipped lexicalized PCFG would clearly have many times the constraints of lexicalized PCFGs alone, which would compound the problem of acquiring a sufficient quantity of training data (see Figure 44). Even if such training data could be produced, it still would not be much help overall, since there is no way for a PCFG to leverage how world-knowledge about one part of a parse tree could affect the structure in a distant part of the tree. As an illustration, consider the sentence *I saw the mouse with a telescope, it was looking to see if the moon was made of cheese*. It is entirely unclear how to compose a set of PCFG production rules to encode that, in this specific case, a mouse is actually using a telescope.

Figure 43. An illustration of how additional knowledge explodes the state space of PCFG Models.

15. Support Vector Machines (SVMs), Perceptrons, and Kernel Machines

15.1 Support Vector Machines

Support Vector Machines (SVMs) (Cortes & Vapnik, 1995)have been increasingly adopted by researchers as an alternative to other kinds of knowledge-lean frameworks. SVMs operate by defining a mapping of input data to a multidimensional space in which the data items can be linearly separated by a hyperplane according to some function of features. These features serve as the criteria that re-map data points in the higher-dimensional space—basically they are an enumeration of the important attributes of the entities being modeled from the perspective of the current reasoning task. Because SVMs rely on the identification of features, they are somewhat akin to Maximum Entropy approaches to modeling. The mapping procedure used by SVMs enables the efficient learning of functions that may not have been difficult to learn if the data was considered in its raw configuration. To accomplish this mapping in a computationally efficient way, SVMs utilize what is known as a "Kernel Trick." In this sense, a Kernel is a vector representation of the features specified in the SVM model. The Kernel Trick uses the properties of matrix multiplication to implicitly calculate the new coordinates of data, thereby bypassing a potentially expensive explicit manipulation. Thus, the popularity of SVMs can be explained by capacity to learn complex functions with thousands of features and their relative efficiency with respect to training.

One of the great boons of the SVM framework is that it is has been shown to also be capable of reducing the dimensionality of input data, which makes further processing even more efficient. Reduction is useful for the SVM framework even though the computational complexity of SVM training is independent of the dimensionality of the feature space, due to the "kernel trick." However, the demands of the algorithm are impacted by the dimensionality of the input space--reduction of the dimensionality of the input space is often required to make difficult inference problems computationally tractable. Moreover, the ability to reduce the number of facets of data that need to be processed during reasoning processes has been shown to be a key attribute of the human cognitive system. There are many examples of this kind of reduction in the various modalities of human cognition. For instance, visual scenes are processed at the higher-

levels after they have been transformed into a much more abstract form of the initial raw sensory input (May, 1998). In particular, it has been theorized that image processing occurs in the human brain as a pipeline that sequentially transforms representations from a raw 2-D retinal image to a 2.5-D "sketch model" and then to a full 3-D internal model of the visual scene (Marr, 1982). This pipeline process great reduces the data's dimensionality (not to be confused with the spatial dimensionality of the internal image model) but progressively eliminating and condensing irrelevant information that has been captured by the senses.

Furthermore, it has been demonstrated that reducing the dimensionality of visual data directly leads to better human performance on related problem-solving tasks (Berg, 2012). Similar trends have been observed in the realm of phonological perception, as well (Hume & Johnson, 2001). Even more complex modes of cognition, such as semantic analysis, can benefit from conceptual mappings that simply the data that is being processed (Lakoff & Johnson, 2008). Lakoff has theorized that there exists a universal set of underlying human concepts that enables the constructions of metaphors that can be shared between individuals via language. Similarly, in the realm of categorical reasoning, grouping disparate individuals into a smaller number of categories such that these categories are centered on some degree of similarity between their member individuals is a strategy that can be exploited to facilitate reasoning. (Iordanova, 2003). While SVMs share this crucial quality of the human cognitive system, to some degree, in practice, SVMs are not capable of approaching the types of reductions that are performed by the human cognitive system. This divergence is caused primarily because SVMs reduce the dimensionality of data in an unstructured and global sense, whereas the reductions performed by the human cognitive system are very structured and apply to specific types of representations and situations, most likely selected for by evolutionary pressures. Thus, it is not possible to claim that SVMs are a solution to human-level reasoning from the perspective of data reduction.

On the positive side, there are computationally efficient methods for performing such reductions using SVMs (Bi, 2003). Nevertheless, there are several reasons why SVMs vastly differ from the human cognitive system in terms of their ability to reduce and process data. Consider a typical study that attempted to make use of the data

reduction capacity of SVMs to facilitate the solving of an NLP problem: (Kim, Howland, & H., 2006). This study involves classifying an input text into one of several possible categories. Text classification using SVMs relies on making a list of terms that appear in the documents to be classified (after pre-processing) and some notion of the semantic relationship between terms that could confound performance (e.g. synonymy and homonymy). The reported accuracies on text classification tasks (including problems involving non-unique classifications of text) seem to indicate that the dimensionality of the input document impacts overall performance. In this instance, however, there are some unrealistic aspects of this reduction, which are representative of general problems of the SVM approach. First, the text classification system relies on a large amount of preprocessing (especially morphological stemming). Second, the number of possible document classes is kept very low. Third, the approach relies on latent semantic analysis (Landauer, Foltz, & Latham, 1998), which has been criticized in its own right, as it relies heavily on overly-simple statistics, such as raw word frequency (Kellogg). As with all statistical NLP methods, SVMs are ultimately incapable of providing general and scalable solutions to the problems that arise while attempting to achieve human-level performance.

Beyond the topic of data dimensionality reduction, there are some major issues with applying the SVM formalism to solving human-level reasoning tasks. Foremost, SVMs are inherently a binary classification algorithm, which means that they can only determine whether or not a given entity has membership in a single category. Many NLP problems, such as PoS tagging, must be cast as a multiple classification problem, because each given entity could potentially have membership in more than one class (Duan & Keerthi, 2006). Since it is clearly inefficient to solve this problem in the trivial way of decomposing all of the membership decision tests into a sequence of separate tests, researchers have developed techniques to adapt SVMs to be able to directly compute multiple classification solutions (Crammer & Singer, 2002).

Even though this fundamental weakness of SVMs has been addressed, there are other more severe criticisms of the SVM approach that transcend the ability to perform multiple classifications. First, the probabilities that are associated with individual items' membership in particular classes are not well-posed by the SVM framework, although

some treatments of this problem have been suggested: (Platt, 1999). It is therefore difficult to use the results of SVMs in a searching framework in which, say, the best n solutions are required. Second, the resulting parameters of SVMs are difficult to interpret; a similar problem is well-known for Neural Networks (Bernatzki, Eppler, & Gemmeke, 1996). Finally, a large-spanning examination of the SVM framework (Bennett & Campbell, 2000) has indicated that the following questions concerning SVMs are currently unresolved: 0.) What are the circumstances in which an SVM will perform better than another kind of model?, 1.) Can SVMs scale up to very large data sets for real-world problems, and 2.) How to use domain knowledge in the design of an SVM. In terms of the knowledge-lean vs. knowledge-rich dilemma, the third question is the most permanent. It turns out that it is the very Kernel Trick that enables SVMs to be so efficient that cripples its ability to incorporate prior knowledge.

15.2 Perceptrons

SVMs are principally known to be a general refinement over the Perceptron formalism for automatic reasoning. Perceptron networks were among the first type of connectionist models that were developed for artificial learning and inference (Rosenblatt, 1957). Although basic perceptrons initially met with success and enjoyed a great deal of excitement from the research community, a serious flaw was eventually pointed out by Minsky and Seymour in their highly influential book, *Perceptrons* (Minsky & Papert, 1969). Essentially, one-layer perceptrons could not learn functions that were not linearly separable (i.e. ones such that a straight line can be drawn on a plot of the data that separates the data into positive and negative examples). For instance, the relatively simple XOR logical function is non-linearly separable. This revelation damaged the reputation of Perceptrons, but this approach regained popularity after multi-layer perceptrons (Cybenko, 1989) were developed that could learn non-linearly separable functions. Analysis of how non-linearly separable functions can be learned by using mappings to higher dimensional spaces lead to the development of SVMs and other Kernel-based approaches. Ultimately, Perceptrons are no more suitable for human-level AI or NLP than SVMs. Like all such methods, they have particular properties that make

them useful for specific tasks, but do not have the ability to scale up to real-world, generalized, human-level problems.

15.3 Kernel Machines

Kernel Machines (or Methods) (Shawe-Taylor & Cristianini, 2004)are a generalization of SVMs, but they do not satisfactorily address the issues with SVMs that are mentioned above. As an example, principal components analysis can be considered a non-SVM Kernel Method (Scholkopf, Smola, & Muller, 1997). Yet again, the development of Kernel Machines represents the type of incremental progress that ultimately will almost certainly not lead to the development of human-level by itself. While it is interesting to study these variations of modeling formalisms to learn about their underlying mathematics, they really are only useful for solving extremely constrained subsets of problems that have been decomposed from human-level intelligent behaviors.

16. Other Statistical, Knowledge-Lean Frameworks

Beyond the approaches detailed in the sections above, there are a multitude of different knowledge-lean methodologies that have been applied to solving NLP problems. All of these methods, when used with their original design parameters fail to capture the complexity of human-level natural language understanding for one or more of the reasons why all other knowledge-lean approaches fail. Since there are so many different approaches and that investigating any of them in any further detail provides little or no insights that have not been gleaned by the analyses above, this section will merely act as a list of references to a selected NLP application that was created with an approach not mentioned elsewhere in this document.

What follows is not intended to be an exhaustive account of all other statistical-NLP methods, but it is a representative selection of them: Neural Networks, which are similar to perceptrons (Collobert & Weston, 2008) (Kazuhiro, Takashi, & Sin-ya, 1992) (Socher, Lin, Ng, & Manning, 2011), Dynamic Associator Networks (Beavers, 2010), Dynamic Systems (Tabor, Galantucci, & Richardson, 2004), Clustering algorithms (Clark A. , 2003), K-Means Algorithms (Hill, 1968), K-Nearest Neighbor algorithms (Li, Shiwen, & Lu, 2003), Second-Order Conditional Random Fields (Sha & Pereira, 2003), Stochastic Gradient Descent (Vishwanathan, Schraudolph, Schmidt, & Murphy, 2006), Latent-Dynamic Conditional Random Fields (Sun, Morency, Okanohara, & Tsujii, 2008), Graph Subsumption (Rus, McCarthy, Lintean, McNamara, & Graesser, 2008), Product of Experts (Das & Smith, 2009), Latent Relational Analysis (Turney, 2006), Vector Space Models (Turney & Littman, 2005), and Toric Grammars (Catoni & Mainguy, 2013).

16.1 Connectionist Approaches

While there has historically been a large amount of focus on the relative strengths and weaknesses of connectionist vs. symbolic approaches to AI and NLP, this only has a small impact on the question of whether or not human-level AI and NLP can be produced via statistical-based modeling. Driven by this question, it makes more sense to think of grouping statistical models not so much based on the connectionist/symbolic distinction, but rather on the data-modeling vs. algorithmic modeling dichotomy (Breiman, 2001). It

has long been known that symbolic, dynamic, and connectionist approaches to reasoning are actually deeply related and can be mapped onto each other with nearly total completeness (Tabor, 2009). Moreover, SVM and Kernel methods, which were discussed above can be seen as a variant of connectionist models (Sun, Appoloni, & Roy, 2002). Like SVMs, Neural Networks can be used to reduce the dimensionality of data to facilitate reasoning over that data (Hinton & Salakhutdinov, 2006). However, the practice of performing such reductions is sensitive to access to large data sets, a large amount of computational resources, and the ability to provide the network with a good initial configuration.

Furthermore, it has been demonstrated that SVMs are actually equivalent to Neural Networks that have been regularized (Andras, 2002). Regularization is a process that mitigates overfitting and stabilizes inducted information for Neural Networks-like approaches (Kudova, 2006). Additional information about the relationship between SVMs and Neural Networks can be found at (DTREG). In all, SVMs are akin to a version of neural networks that enable the "kernel" function to be explicitly defined. This property has a clear advantage for the development and maintenance of models, so there are really no special attributes that make traditional connectionist approaches any better than SVMs.

The use of algorithmic models, however, may be problematic in and of itself, as there is no real control that researchers have over the exact algorithm that is produced as a result of learning. Instead, it can only be expected that a learned algorithm acts as a map between the input and output data. Many different mapping functions are consistent with large data sets and it can be difficult to determine a priori which of possibilities are "truer" to the function that actual created the data set. Certainly, if the idiosyncrasies of the human cognitive system are important to its function (and there is evidence that this is the case (Marcus G. , 2008)[63] (Gigerenzer, Todd, & Group, 1999)), then it would be unwise to use algorithmic learning processes that ignore the structure of that system. On the other hand, all existing data modeling formalisms are far too simple to represent all of the types of processing that occur in the human brain. A succinct treatment of the

[63] Although see (Perlovsky, 2010) and (Bringsjord & Bringsjord, Kludge: The Haphazard Construction of the Human Mind, 2012) for contrary opinions.

shortcomings of Neural Network architectures can be found in: (Hoffmann, 1990). Among the main points of this presentation are that Neural Networks require specifically tailored construction for individual problems and that there is a need to invent more complex network topologies for complex problems, since representational power is fundamentally limited by network complexity. Both of these issues seem to demand the kind of expert-knowledge-based, direct intervention that statistical approaches were intended to avoid. And yet, bare statistical approaches are grossly underpowered for human-level tasks; indeed, it is a general law in modeling cognitive processes that simple configurations are unable to handle complex phenomena.

Nevertheless, in the spirit of completeness, there are two elaborations of connectionist approaches that are worth exploring briefly. First, a type of connectionist-like approach called a Dynamic Association Network (DAN) has been developed as a model that can extend its own state space (Beavers & Harrison, 2012). One of the major flaws of traditional connectionist approaches is that they are static, as it seems unlikely that complex, biological neural networks would be constrained in this way. Despite the advance that DANs represent, they still falter from an inability to learn and leverage the large amounts of knowledge that human-level NLP requires. Some promising avenues for future work in the vein of DBNs have been suggested, namely work that involves teleodynamic emergent systems (Deacon, 2011).

Second, deep learning algorithms (Deng, Dong, & Platt, 2012) for Neural Networks have been devised in order for the structure and operation of these models to resemble the human cognitive system to a greater degree than standard Neural Network-like approaches (e.g. some models are configured to approximate the structure of human cortical regions (Poon & Domingos, 2011)). Not only do deep architectures have tractable learning algorithms, but they also can be shown to outperform both standard Neural Networks and SVMs on some tasks (Salakhutdinov & Larochelle, 2010). However, the most important aspect of deep Neural Network architectures is that they are capable of implementing inference algorithms that rely on a metric of similarity between data items (Salakhutdinov R. , 2009). This type of processing is required for analogical reasoning, which is thought to be one of the backbones of human cognition. Yet, the basic kinds of analogies that deep Neural Networks are currently able to support pale in

comparison with the complexity of the mappings that occur in the human mind (Wagenmakers, Farrell, & Ratcliff, 2005). Self-organizing systems of all kinds have difficulty producing the kinds of rich interactions and structures that are currently thought necessary for human-level cognition.

Other self-organizing algorithms, such as self-organizing maps, are a related connectionist approach to cognitive modeling that attempt to exploit the ability for structure to emerge from an initially plastic network (Ultsch & Siemon, 1990). Again, this type of approach is motivated by the belief that emulating the operation of the human brain would be useful for creating human-level AI and NLP applications. While this is undoubtedly true, it can be argued that these approaches are attempting to capture the wrong level of structure in the brain. It seems as though such methodologies ignore the presence of top-down influences in human cognition. Although it is entirely possible that such feedback loops emerge spontaneously from low-level neural networks, the details of whatever processes that accomplish this are almost entirely unknown. At this point, it is therefore inadvisable to believe that relatively basic models of the current interpretation of the structure and function of the human cortex are capturing just the right relationships that give rise to human-level cognition. While it is probably necessary to study the low-level processes in the brain, at this point, higher-level processes will probably have to be engineered directly based on macroscopic analyses of human cognition. Such an endeavor will almost certainly require the use of knowledge-rich modeling paradigms.

16.2 Generalized Improvements to Statistical Approaches: Ensemble Learning and Boosting

The drawbacks of individual statistical modeling assumptions and paradigms have led to work towards the development of general-purpose framework-independent algorithms to increase the performance of machine-learning techniques. There are several different methods that can be applied to machine-learning algorithms that improve their performance in a general way (i.e. they are not dedicated to improving any individual formalism. However, for the purposes of this document, it is only necessary to give a cursory overview of two of the most widely-used methods: ensemble learning (Zhou, Wu, & Tang, 2002) and boosting (Kearns M. , 1988), because they have little

impact on solving the human-level NLP problem. While such methods are capable of improving the accuracy of machine-learning approaches for a large number of important applications, ultimately they are not conducive to the goal of achieving human-level NLP. This is not to say that there is absolutely no utility to exploring these methods, but only that, at best, they represent a distraction in the quest for human-level NLP. Like many of the approaches described above, ensemble learning and boosting techniques are capable of producing small, incremental improvements on existing approaches, but fail to correct any of the underlying shortcomings of those platforms. Thus, they ultimately become a distraction that absorbs resources that could be better applied elsewhere.

To see why this is the case, first, consider ensemble learning. Ensemble Learning is the practice of applying multiple methods to find a solution to a problem and then use the combined results from those methods in the final analysis, in the hope that this will provide more complete coverage than any of the individual approaches alone. Clearly, if all of the approaches used in an ensemble learning scheme are incapable of representing fundamentally important aspects of a problem, and since no knowledge-lean approach has been demonstrated to overcome the general deficiencies of statistical NLP, then there is no reason to believe that combining any number of those approaches will spontaneously result in a vast shift in performance. By analogy, this is akin to trying to fill a cube if all that is available is different coverings for its surfaces. More advanced kinds of ensemble learning, such as bagging (Breiman, 1996), are specifically designed to help reduce problems with overfitting to data and to differences in results when different data sources are used for training and evaluation. Although these qualities are obviously helpful for resolving some of the minor issues with the corpus-based approach, they do nothing for the more integral matters, such as the need to combine world knowledge with information about grammatical constructions to create a human-level parser.

On the other hand, boosting methods attempt to overcome some of the downsides associated with ensemble learning by assuming that a set of simplified learners can exhibit a kind of emergent complexity. As an example of why boosting may be useful, consider that certain methodologies, such as HMMs, make the assumption that a given data set obeys a particular probability distribution, such as a Gaussian Mixture. If the true distribution properties of the data are unknown and an incorrect assumption about

the distribution is made, then the accuracy of any resulting model is likely to be relatively low. Boosting is one method for overcoming the imposition of assumptions on the distribution of data and the need to model the entire joint distribution of evidence and classifications. By combining a number of very simple classifiers, usually formalizations of very simple heuristics, boosting overcomes the biases of overly-specified modeling formalism. It may be argued that boosting is similar to the type of mathematical modeling used in Maximum Entropy models, only that a boosting model is decomposed into weighted learners instead of weighted features.

However, as opposed to ME models that use features to maximize likelihood, boosting procedures use features to define an abstract hyperplane in data feature space that provides an optimal partitioning of data items in that space. But, this makes boosting algorithms similar to SVMs, which have their own difficulties. Like SVMs, boosting was initially developed for binary classification (Freund, Schapire, & Abe, 1999), but eventually was extended to handle the multi-class case, which would be required for most standard NLP tasks, such as PoS tagging or language identification. Despite this advance in boosting procedures, they are ultimately unfit for bridging the gap between current knowledge-lean approaches and human-level NLP. This should not come as a complete surprise, since boosting is closely related to Support Vector Machines in a mathematical sense (Zhang & Gu, 2011), which implies that they would falter along at least some of the same lines.

17. Summary of the State of Statistical NLP

Through the review of statistical NLP approaches given above, there are some prominent trends that can be identified for this body of work as a whole. Primarily, it is the case that many of the technical and qualitative shortcomings of these approaches are known and have been addressed individually. The sheer number of different species and subspecies of models currently employed, coupled with the ongoing continuous development of refinements to these methods, indicates that none of these approaches alone are meeting the mark demanded by the end users of NLP systems. While it may be unfair to judge the statistical approach by the fact that it has not yet achieved human-level performance, there are some other criteria that cannot be avoided in a final evaluation. It is true that there are some well-performing NLP applications that make use of knowledge-limited data, but these are geared for either very specific tasks or tasks in which the expectations are low. Since there is a very high demand for better NLP, it would almost certainly be the case that this demand would be met by some organization if it were possible to do so. This is probably the most condemning observation that can be made about the prognosis of the NLP field as it currently exists.

From the perspective of individual modeling frameworks, a number of common themes arise when attempting to explain why they are unfit for creating human-level NLP systems. First, all of the methods described above are incompatible with the context-sensitive, long-distance dependencies that are extremely common in natural languages. There are many ways to manipulate the basic statistical-learning formalisms to expand their representational power, but this always results in some kind of unacceptable trade-off, such as loss of efficiency, a need for approximation algorithms, or a need for much larger amounts of data. Second, most of the methods rely on corpus-based learning methods, which are fundamentally incompatible with the knowledge-rich features of natural language understanding. On one hand, there are cases in which commonsense or world knowledge is required to interpret text that is present in a corpus. On the other hand, knowledge can be required to fill in words or phrases that have been elided from a corpus. There are few if any approaches to solving ellipsis at the human-level. Third, evaluation metrics and modeling formalisms tend to fit each other in a self-reinforcing loop. Models are created that will perform well under the most common evaluations and

evaluations are created that are compatible with the most common types of models. This reduces the chance that researchers will explore in the correct direction to achieving human-level performance, as progress towards that goal has hardly any incremental reward given the current research environment.

Fourth, although knowledge-lean approaches to not require information to be directly coded into them and thereby avoid some issues with flexibility and brittleness, they are incapable of significantly modifying themselves after their initial parameters have been set. Because of this, knowledge-based approaches either must be trained for very specific types of data sets and task definitions or they must be hand modified by humans in order to be purposed for applications beyond their initial design. The fact that new knowledge-lean models are usually trivial to create once the underlying formalism has been discovered, there is no inherent penalty for researchers to merely generate new models whenever that is required. Fifth, knowledge-lean approaches do not have adaptable inference algorithms. This prevents them from being able to perform belief revision during the reasoning process, for instance. For example, it might turn out that a decision made late in the parsing process should change the likelihood of a previous decision. Knowledge-lean approaches are not capable to accommodate such revisions, even for something as simple as trying to revise a parse to contain a verb if no verb is assigned by the end of the current sentence.

Sixth, knowledge-lean approaches must create naïve estimates for patterns that are not encountered in training data. Lexicalized methods have a tremendous difficulty with this because there are so many combinations of lexicalized structures that most of them would never occur in even extremely large corpora. Also, consider that Maximum Entropy approaches attempt to make no judgments about aspects of the data for which they cannot directly account. Humans continuously make predictions and assumptions via analogical reasoning. Even though humans may be biased into making unwarranted assumptions, they still are capable of handling most kinds of novelty far better than any existing knowledge-lean approach. The book *Fooled by Randomness* (Taleb, 2005) contains a large amount of advice for adaptive reasoning in the financial markets environment. Some of the greatest mistakes humans make when attempting to emulate formal reasoning systems is discounting the importance of rare events and focusing

entirely on frequency to make probability judgments rather than incorporating world knowledge as well. Both of these mistakes are routinely made by knowledge-lean systems. Furthermore, the book touches on the problem of induction, which illustrates how reasoning can go awry if a framework of knowledge is not imposed on the inference process. Consider the situation of an agent observing the rise of the sun each day. Over time, the induced probability that the sun will rise on subsequent days increases monotonically. Intuitively, humans know that this cannot be the case. The reason for this belief is that humans have knowledge that the sun will one day burn out and therefore the probability of the sun continuing to rise should be conditioned on that information, as well. Therefore, without some knowledge, the soundness of reasoning is thrown into doubt.

17.1 Possible Explanations for Failure of Statistical Approaches

Although humans do not necessarily comprehend sentences at a better rate, there are some important considerations to make in light of this revelation. First, humans are able to proceed forward with parsing and discourse comprehension, even if processing fails at a lower level. This is due in part because humans do not segregate their processing into discrete stages, but rather utilize top-down strategies, such as maintaining expectations, to help resolve ambiguities as they are encountered. Second, statistical-based approaches almost always have difficulty accommodating to multiple different distributions of input data. This occurs mainly when the corpora used in training models differ from the corpora on which they are evaluated (Ratnaparkhi, 1999). This issue is compounded by the fact that there are few large-scale research corpora that are tagged for use by supervised learning techniques (Rennie, 2003).

Several approaches aimed at improving the generality of NLP systems have resulted in greater successes at transferring learning between domains. However, an examination of the state-of-the-art performers would indicate that few of the highest-performing methods incorporate the features required to transfer learning between significantly different corpora. In comparison with human language acquisition, there are two facts to consider. There is apparently a finite "critical period" for humans in which they must be exposed to language in order to realize their full linguistic potential.

Beyond this critical period, the maximum level of attainable proficiency falls rapidly (Purves et al., 2001). Also, infants quickly lose the ability to perceive phonemes that do not occur in the languages spoken in their environments. These facts suggest that there are some aspects of language are not adaptable after a certain point, which is somewhat in line with trying to engineer an NLP system that has a discrete learning phase. On the other hand, humans clearly are capable of learning linguistic knowledge, such as foreign languages and additional vocabulary, throughout their entire lives. Non-adaptive knowledge-lean approaches therefore can therefore not capture the complete operation of the human cognitive system. On a smaller scale, humans often will reanalyze sentences that are ambiguous or unusual. A prime example of this is the process through which garden path sentences are understood (Ferreira, Christianson, & Hollingworth, 2001). Such behavior directly contravenes the operation of knowledge-lean approaches that are fundamental incapable of belief revision. Furthermore, people use expectations to understand linguistic constructions that they might not have previously encountered (Altmann & Mirkovic, 2009), which serves as another form of adaptation.

Some researchers have made attempts to formalize the intuition that natural language processing involves more than just patterns of words or even patterns of phrase structures. (Schank, 1972). If this theory is true, then knowledge-lean systems that learn on corpora that consist entirely of word patterns have no hope in capturing the intricacies of human language. Even if alternative corpora could be produced, statistical NLP approaches would still be trained to exhibit the "average" case without learning the exceptions and syntagmatic and paradigmatic patterns of expectations that can only be derived by more fine-grained analysis. The mapping of NLP structures to discrete abstract categories is almost certainly a lossy simplification for the process that humans use to compile larger units of meaning from smaller units.

Indeed, unlike most NLP methods, humans do not need to totally resolve ambiguities in language to obtain a good understanding of the intention of a message, which implies that there are larger structures of communication at play than individual phrases or words. During language comprehension, humans cordon off such ambiguity for as much and as long as is possible (Blache, 1998). In some cases, ambiguity is purposefully created for social tact (Lifestyle Lounge), humor (Oaks, 2012), or other

rhetorical purposes (Quinn, 1993). Such types of language behavior that go beyond the actual lexical content of the utterance, including declarations of one's personal state, commands to others, and promises, are known as illocutionary acts (typical language use would be considered to be a locutionary act) (Austin, 1962). Since this type of information is not encoded in current corpora, and some of it may not ever be practically encoded, the machine-learning techniques that rely on those corpora are forced to ignore this fundamental aspect of human language.

There are even deeper questions about whether or not the knowledge-lean approaches are actually accomplishing something useful in terms of human-level NLP. This can be illustrated by examining what might happen to the statistical NLP field if linguistic theory is significantly changed from the present mainstream understanding. As an example, some linguists are skeptical that definite categories for lexical entries (e.g. words) exist or are even a helpful approximation (Barner & Bale, 2002). Consider the possibility that this is true: then a large number of long-established NLP problems and approaches would be rendered obsolete. Clearly, PoS tagging would be the most directly impacted, as that task would become meaningless under the new regime. One might initially think that this is not too severe of an issue, because many of the more "interesting," higher-level NLP problems would seem to be preserved. Intuitively, the idea that language concocts meaning from syntactic and semantic manipulations of and compilations into ever more abstract structures is so sensible that there are such things as predicates at some level even if individual words cannot reliably be mapped to parts of speech.

Even if there is not truly a stable division between nouns and verbs, then it is conceivable that parsing could still be performed in the case that noun phrases or verb phrases are no longer coherent concepts (i.e. a different phrase structure system might have to be invented). Yet, this would potentially provide a major challenge for the knowledge-lean NLP community. One issue is that the PoS tagging task is often used as a benchmark to vet specific statistical formalisms. PoS tagging, even if it does not correspond to psychological reality, certainly gives some idea about how efficiently and accurately an algorithm can extract information. However, success on this type of assessment might only be correlated to the activities of a human-level NLP system.

Without insight into or a theory about how a human-level NLP system should work in general, it is unknowable whether a given approach is modeling something central to the operations of a natural language processor or if it is merely reflecting some arbitrary projection of the operations of such a system.

Although knowledge-lean approaches could certainly be rapidly configured to adapt to a new linguistic theory, this would not eliminate the possibility that such approaches are useless for achieving human-level performance, at least in a computationally tractable way. It is very awkward for statistical approaches to incorporate comprehensive, knowledge-rich theories and structures without greatly decreasing their efficiency and robustness. For instance, it might turn out that all languages, not just agglutinative ones, would require joint syntactic parsing and morphological analysis to achieve human-level performance. This would almost immediately obviate the used of some kinds of modeling formalisms. If semantic theories or commonsense reasoning were also required for parsing, then knowledge-lean representations and their training corpora would be useless for this task. At the very minimum, new corpora and evaluation standards would have to be compiled. Because relative performance low-level tasks is a major influence on how much attention a given approach receives, the results from these redesigned experiments might have an enormous impact on the direction of the field. Since knowledge-lean approaches are often hand-crafted and optimized for specific tasks, they lack the ability to generalize beyond their initial assumptions.

Thus, it is entirely possible that approaches currently in play in the knowledge-lean arena are inherently unsuitable for human-level NLP. There is no way to know this without changing the research program in this field from one of solving artificial, isolated, machine-learning-friendly problems. It has been said that the most important data to model in AI is that humans are capable of human-level intelligence (Cassimatis N. L., 2006). From this, it follows that the most important data to model in NLP is that humans are capable of human-level NLP. Without taking this as the principal goal of the NLP field, it is not likely that satisfactory artificial systems will ever be created.

Examining the relative performance of different statistical NLP algorithms can also be useful for understanding the weaknesses of this methodology. In a paper about

unlexicalized PCFG parsing (Klein & Manning, 2003), it was revealed that unlexicalized parsers perform nearly as well as lexicalized parsers (provided they have phrasal type labels) and that modifying lexicalized parsers to ignore bigrams barely influences their performance. These facts are probably indicative of the fact that only a limited amount of the structure of language is explainable via CFG-style parsing by learning from bracketed text and that the current state-of-the-art has probably tapped out all of the potential that can be obtained from such methods. Some of this information seems to be corroborated by the statements in (Manning & Schütze, 1999) implicating the impact of the Penn Treebank bracketing practices for the relatively low performance gap between lexicalized and unlexicalized parsers. Thus, it seems fair to assume that such methods are not really learning anything about natural language, per se, but are instead learning some abstract function that is correlated to structures in natural languages. It is unlikely that the algorithms that are specifically tailored to be successful at modeling those functions would also be adept at modeling the functions that actually underlie human-level NLP.

Finally, one can appreciate more of what the true role of machine-learning approaches to NLP should play in a system that is designed to perform at the human-level. In a paper about ambiguity resolution (Franz, 1996), one major flaw of the original knowledge-rich approaches to NLP is pointed out; namely, that they failed to account for important syntactic generalizations that are widespread in natural languages. However, there are many common patterns that data mining, statistics-based approaches to NLP uncover readily from corpora. Since natural languages at least somewhat adhere to licensed patterns of constructions (at least in reasonably grammatical speech), ignoring them results in the loss of a great amount of regularity that might help reduce the combinatorial problem of having to hand code knowledge-engineered rules individually for every specific case. However, aside from this issue, knowledge-rich approaches address all of the shortcomings of knowledge-lean approaches. It appears that rather than having separate knowledge-rich and knowledge-lean methods, researchers should focus on combining them. Based upon all of the analysis above, the best configuration of a system would seem to be one that utilized the adaptive, belief-revising inference and complex representations of knowledge-rich systems while using the pattern learning

capability of knowledge-lean systems to automatically expand the capabilities of the core knowledge-rich implementation.

18. Forward Directions

From this extensive analysis, several overall points should now be clear: 1.) the problem of human-level NLP is relevant to both general AI and Cognitive Science, 2.) there are many scientifically and financially lucrative applications for sophisticated NLP systems, 3.) statistical approaches have largely supplanted knowledge-rich approaches in the NLP field, and 4.) statistical approaches are not nearly as effective as they are purported to be and are unlikely to result in sufficiently powerful NLP systems for the target applications. The initial question of why statistical and knowledge-rich approaches to NLP (and AI) both fail to capture the nuances of human-level has been addressed. Having established the main reasons why the reported effectiveness of statistical NLP is actually false, the desire to actually build human-level NLP systems motivates an inquiry into what, then, would be a promising alternative avenue for future NLP research. Thus, there is another question to be answer, which is what can be done if the two major methodologies of AI research apparently are both ineffective.

18.1 Unification of Statistical and Knowledge-rich AI

One of the major conundrums of the above analysis is that it appears that the two options—either knowledge-rich or knowledge-poor—for creating human level NLP systems are unfruitful. Indeed, statistical and non-statistical approaches are both vulnerable to failure because it turns out that they are fundamentally equivalent. Research into hybridizations of logic formulae and graphical models yielded the Markov Logic formalism, which is explained in detail in: (Domingos, Kok, Poon, Richardson, & Singla, 2006). Ideally, this information should enable some kind of system to leverage the strengths of the knowledge-lean and knowledge-rich approaches against their respective weaknesses. However, Markov Logic, as described above, fails to meet the mark required for human-level performance by a large margin. Knowing this, it is possible to examine why the shortcomings of Markov Logic can point to the best directions for future research that aims to accomplish human-level NLP.

Markov Logic does not succeed because it improperly addresses the major shortcomings of statistical and non-statistical processing. At a very high level, knowledge-rich systems fail since they must be engineered by hand because there are no

efficient algorithms or sufficiently large corpora to train them. On the other hand, knowledge-lean systems are far too simple and are extremely rigid after their initial parameters have been set, as they lack any ability to perform complexly structured inference. Because of the way that knowledge-lean and knowledge-rich aspects of reasoning are combined within the Markov Logic framework, a state-space explosion occurs after models expand beyond a fairly small size. This phenomenon occurs because it is impractical to attempt to learn probabilities for every possible state that can be grounded out from the initial Markov Model description. Recall the example given above of this simple rule that grounds out to 10, 000, 000, 000 constraints with only a relatively few number of objects (at least far fewer than might be encountered during a real-world reasoning problem):

$$Location(?x1, ?p1, ?t1) \wedge Location(?x2, ?p2, ?t2) \wedge \neg Same(?p1, ?p2, E) \wedge \neg Same(?x1, ?x2, E) \overset{(\infty)}{\Longrightarrow} \neg Same(?t1, ?t2, E)$$

Figure 44. An Example of a Constraint that Would Need to be Grounded in a Markov Logic Model.

This grounding process is mandatory for Markov Logics, so it is necessary for approximation algorithms and unstructured inference algorithms to be used in order to make reasoning in this framework tractable. Unfortunately, this merely serves to shunt some of the worst shortcomings of purely knowledge-lean systems onto the solutions for these issues of efficiency.

Ideally, a solution to the problem of how to properly combine knowledge-lean and knowledge-rich methods would have the following properties. First, it would be capable of learning in an efficient manner. Second, it would be capable of modifying itself significantly without human intervention after its initial configuration has been set. Third, it would take all of the highly nuanced and context-related knowledge-based clues to solve problems that cannot be solved by the results of training on data that contains none of that kind of information. And forth, it would involve some kind of function for mapping data from many different domains into decomposed problems that could be solved by just a few special-purpose processors. It turns out that there is a kind of

platform that is capable of exhibiting all of the above characteristics—it is called a Cognitive Architecture.

18.2 Cognitive Architectures and the Human Cognitive Substrate

Cognitive Architectures (Newell, 1994) are an approach to creating broad platforms for (potentially) general-purpose reasoning systems. Rather than provide a single solution to a single problem, Cognitive Architectures provide a framework within which numerous models can be created for a variety of different tasks. There are many well-known Cognitive Architectures, each of which is geared towards different types of theories on human cognitive inference, and representations. To determine whether Cognitive Architectures meet the criteria that were suggested above for a human-level potential NLP system, all that is necessary is to show that the various components of such a system can be implemented in a Cognitive Architecture.

First, Cognitive Architectures must have the capacity to learn and to self-adapt to new information. This quality is exemplified in the ICARUS architecture with models created for studying the learning of hierarchical skills for solving problems in BLOCKSWORLD and an inner-city driving environment (Langley, Choi, & Rogers, 2005). The research performed by the ICARUS group involves interleaving learning and inference, which is not something that statistical approaches can accomplish explicitly. Through interleaving these processes, it is possible to simulate the same kind of push-pull, top-down-bottom-up reasoning that appears to be implemented in the human brain. Second, Cognitive Architectures must have the capacity to integrate disparate types of information into a unified framework. This quality is exemplified in the ACT-R architecture and its various extension modules (Anderson J. R., 2004).

Third, Cognitive Architectures must have the capacity to combine different types of algorithms in the same framework to exploit their respective strengths while simultaneously mitigating their individual weaknesses. This quality is exemplified in the connectionist and symbolic aspects of the CLARION architecture (Sun R. , 1997). Moreover, the Polyscheme Cognitive Architecture generalizes this kind of hybridization to seamlessly incorporate arbitrary kinds of reasoning algorithms (Cassimatis N. L., 2007). Fourth, Cognitive Architectures must have the capacity to implement an inference

algorithm that can adapt to new information. This quality is exemplified in the SOAR architecture's means-ends search-based approach to reasoning (Rosenbloom, Laird, & Newell, 1993). A sixth desirable characteristic of a Cognitive Architecture is that it provides a well-motivated mapping between arbitrary problems in many domains to more stereotyped problems in a few domains. The Polyscheme Cognitive Architecture (Cassimatis N. L., 2006), in addition to supporting general hybridized algorithms, implements a cognitive substrate theory that maps problems in all domains to physical and social reasoning problems.

It turns out that this sixth quality is essential to the task of creating human-level NLP systems. The ramifications of this kind of mapping between problem representations was discussed in the above sections on SVMs and connectionism as a way to reduce the dimensionality of the data that are processed during a particular instance of reasoning. Reducing the dimensionality of knowledge-rich data is also essential to creating learning methods and corpora of automatic learning. If annotations have to be produced in all of the possible domains with which a piece of information could be involved, then the learning problem quickly becomes intractable. However, if data can be distilled down into components that belong to only a few domains, then it becomes possible not only to have efficient learning algorithms, but also to have the ability to produce corpora from which that learning could take place. On the other hand, this kind of mapping would allow both linguistic and non-linguistic knowledge to be handled seamlessly inside of the same reasoning framework, which is obviously desirable for hybridizing knowledge-rich and knowledge-poor methods. For instance, broad patterns in language could be learned by knowledge-poor methods, while more specific semantic and pragmatic theories could be captured by knowledge-rich models.

Once these two processes are seeded, they can interact to better each other in a positive feedback loop. Semantic information can be used to improve learning at the lower levels and information about common syntactic patterns can aid in the interpretation of language. Furthermore, if this mapping is based on high-level features of the human cognitive system itself, then there is more reason to believe that the resulting algorithms that emerge for processing language will be similar to the actual algorithms used by the human brain. Recall that the algorithms that are learned by

algorithmic modeling statistical formalism are much more arbitrary and would be decoupled from how the human cognitive system functions. Also, recall that it has been hypothesized that the particular idiosyncrasies of human reasoning have survived evolutionary pressure because they represent assumptions that enable real-world reasoning problems to be solve more effectively.

While the six elements suggested above are present in individual architectures, it is necessary to incorporate them all into a single Cognitive Architecture so that they can all be brought to bear on the problem of solving the human-level NLP problem[64]. Fortunately, the Polyscheme Cognitive Architecture offers the capacity to support this kind of integration. In fact, all of the above requirements have been met in Polyscheme with the exception of combining a knowledge-lean learning algorithm in a model. However, the integration of new modules is so easily handled by Polyscheme that at least making a first pass at doing this would be straightforward. Integration of a new module in Polyscheme requires that an algorithm be decomposed into a set of common functions. These common functions communicate using abstract, representation-independent propositions, which can be translated back into the internal representations used by each of the modules themselves. To integrate a HMM into Polyscheme, it would essentially require only that two functions be created: one that would map propositions to representations of data (i.e. a word and a tag) that the HMM could directly use and another that would perform the inverse mapping. Although there are almost certainly more sophisticated approaches to this kind of integration (e.g. enabling results from other modules to interact with the HMM more directly), the simple procedure described here suffices as a first step, proof-of-concept.

It is useful here to briefly digress to examine in more detail Polyscheme's ability to support hybrid algorithms. Reasoning in Polyscheme is in part centralized and in part distributed. The centrality aspect of the Polyscheme Cognitive Architecture is necessary from the perspective of needing to create adaptive hybrid algorithms. Algorithms can be hybridized if they are broken down into individual time slices that can then be distilled down to a small number of discrete stages. For each time slice, the individual algorithms begin to run their stages in order. Because these algorithms can run in parallel,

[64] And thereby providing valuable insight into methods for solving the human-level AI problem, as well.

information about the state of the other algorithms can be shared via the central Polyscheme interface. Sharing information requires that the representations used by each of the algorithms be translated into a common representation. Polyscheme uses a data structure called a proposition, which is composed of a predicate, arguments, a time object, and a world object. For instance, a proposition that represents the knowledge that time A is before time B at all times E and in the real world R[65] might be formulated as:

$$BEFORE(A, B, E, R)$$

Figure 45. An example of a Polyscheme proposition.

Because the operations of the component algorithms must be synchronized, Polyscheme maintains one proposition in focus during each time slice. This focus is associated with a particular type of attentional modality that indicates how this particular proposition should be treated by the individual algorithms. Examples of these types of attention are focuses to assert or deny the truth of a proposition or to speculate what are the ramifications of a particular proposition being either true or false. Each focused proposition is transmitted simultaneously to every algorithm, at which point the information the proposition contains is translated back into the dedicated representations used by that algorithm. Ultimately, each algorithm produces an opinion on the current proposition in focus and makes suggestions about what propositions should be focused on in the future. These opinions provide the level of positive and negative evidence for the truth of the proposition in focus that each algorithm has calculated. In this way, parallel algorithms, even if they are radically disparate, can be harnessed to work in unison on the same reasoning problems.

From the distributed perspective, Polyscheme admits the definition of Specialists that act as a way to encapsulate domain-specific representations and algorithms from the other Specialists that are defined in a given implementation of the architecture. To ensure that Specialists are not created superfluously, they must be motivated by some aspect of reasoning that in involved with many more abstract domains and also that cannot be decomposed into other specialists. Specialists ultimately must communicate

[65] The "real world" here is considered to be a logical world that has no assumptions in its basis.

information to each other, so it is necessary for them to implement the same abstract functions. These functions are known as "common functions" and it is believed that through the use of these functions, essentially any useful algorithm can be decomposed into a procedure that can be interleaved with information from algorithms in other Specialists that are running in parallel. The most fundamental common functions of a Specialist are: $FOCUS$ (communicates the current proposition in focus and the attention modality that is associated with it), $OPINION-ON$ (the current opinion of the Specialist), $REPORT-OPINIONS$ (a function to report the opinions of other specialists to the Specialist), $GET-ATTRACTIONS$ (the propositions that the Specialist believes should receive focus in the future), and $FINALIZE$ (a function to report the final result of inference from the current time step):

$Specialist\ Activity\ Per\ Cycle$:

$$FOCUS(propostion, attraction)$$
$$opinion: OPINION-ON()$$
$$REPORT-OPINIONS(opinion[])$$
$$opinion: OPINION-ON()$$
$$\{proposition, attraction\}[]: GET-ATTRACTIONS()$$
$$FINALIZE(opinion)$$

Figure 46. The time sequence of Polyscheme Specialist inference.

For every time slice, all of the common functions are executed in order for each Specialist in parallel. Inference is complete once all propositions that have been marked for focus have been focused on. At the end of inference, the solution to the problem posed to Polyscheme can be considered by combining all of the opinions from all of the Specialists on all of the propositions that have been in focus.

Furthermore, it is also useful to examine Polyscheme's theory of the cognitive substrate to better understand how language can be mapped onto physical and social reasoning. There are sound reasons from an evolutionary perspective to believe that human cognition is largely geared to solving problems couched in these modalities and studies of abstract thinking have demonstrated that most, if not all, metaphorical thought can be understood in terms of physical concepts (Lakoff & Johnson, 1980). Accepting this hypothesis, however, does not provide an exact definition of the human cognitive

substrate. This is the point where the Polyscheme cognitive substrate theory becomes involved. After a wide-spanning study of human reasoning processes and previous work in this area, the following major domains were identified as the most likely members of the human cognitive substrate: time, identity, causality, space, categories, simulations (i.e. worlds), containment, and reification/creation of objects.

Each of these domains is represented by a Polyscheme Specialist that is associated with efficient and encapsulated representations and inference routines. Allen's Interval Calculus (Allen, 1983) is an example of the types of reasoning formalisms that support inferences in these domains. In terms of representation, periods of time are represented as intervals that themselves are comprised of a start point, a mid-point, and an end-point. In terms of inference, Allen defines thirteen of possible relationships that can hold between different intervals (e.g. time *A* is before time *B*). It turns out that it is possible to represent all of these temporal relationships by decomposing them into "before" and "equals" relationships that hold between the start and end points of intervals. A graph-based searching algorithm can be used to calculate the evidence for and against a particular relationship between two times given the current state of knowledge about the relationship between other intervals.

The strength of the Polyscheme approach is that these basic, domain-specific processors can be compiled to produce extremely complex behavior without explicitly encoding that behavior. For instance, it has been demonstrated that Polyscheme Specialists can be utilized to implement a DPLL-like algorithm that can support search during the model inference process (Cassimatis, Murugesan, & Bignoli, 2009). Ultimately, algorithms of ever-increasing complexity can be produced by continuing to combine the basic substrate elements. An in-depth examination of how natural language understanding can be mapped to the Polyscheme cognitive architecture can be found in: (Cassimatis N. L., 2004).

In brief, words, phrases, and sentences can be considered to be events that require an explanation. The explanation of a word is the existence of a phrase of which it is the head; the explanation of a phrase is the sentence to which it belongs, etc. Alternatively, words and phrases can be considered to have extensions in time, which can be used to begin to impose deep structure on the emerging parse. The types of phrases can be

considered to be categories and constraints (implemented by causality statements) can be written to assign words to the possible phrase categories that they could belong to given their parts of speech. Furthermore, reasoning about identity can support reference resolution. Since there are multiple ways to parse a sentence, Polyscheme can use its capacity to search to evaluate different parsing possibilities in an attempt to find an optimal solution. Through this type of reduction, it is possible for Polyscheme to represent and reason in a knowledge-rich way without requiring arbitrarily complex domain specifications.

18.3 The Role of Hybrid Algorithms in Cognitive Architectures

The practice of using hybrid algorithms to work towards achieving human-level NLP (and human-level AI in general) is extremely important and thus deserves to be considered in some further detail here. Not all cognitive architectures are truly capable of algorithm hybridization, which is why Polyscheme was selected as the primary focus for this section of the dissertation. To be clear, hybrid algorithms are such that they enable vastly different kinds of inference procedures and representations to seamlessly be brought to bear on the same reasoning problems. This kind of procedure is necessary for overcoming many of the difficulties faced by algorithms that are not designed with this integration in mind. From the above analysis of methods within the knowledge-lean and knowledge-rich paradigms, two common modes of failure have become clear. First, both knowledge-lean and knowledge-rich approaches suffer from a lack of adaptability without intercession from human agents. Second, both knowledge-lean and knowledge-rich approaches both suffer from state space explosions under complexity.

In terms of the lack-of-adaptability concern, algorithms that are not only hybrid, but also adaptive algorithms are necessary, which is another justification for the use of Polyscheme.

One possible solution to these problems is to try to eliminate as much of the complexity as possible from reasoning problems in order to allow non-hybrid algorithms to achieve quantitatively good results. However, as has been illustrated above, this practice results in the adoption of deceptive metrics and/or problem definitions that are so narrow that obtaining a high accuracy on them is essentially irrelevant for the general

NLP problem. Moreover, many studies in Linguistics and Cognitive Science all point to the conclusion that a high level of complexity is needed to achieve human-level performance on NLP tasks. The question of how to efficiently handle this level of complexity therefore remains. From past research in Polyscheme, the best answer currently appears to be that reasoning systems can only marry large amounts of structured world-knowledge with computationally efficient processing by combining slow, expensive specific-purpose representations and algorithms with more efficient general-purpose ones. Heretofore, Hybrid Cognitive Architectures appear to be the single type of platform that can support this combination. Details of how Polyscheme enables this kind of integration can be found in Chapter 19 and Chapter 20 below.

It turns out that prior research is able to provide a glimpse into the power of using approaches that are able to directly combine specific and general representations and algorithms. In terms of sheer performance improvements, one of the best illustrations of Polyscheme's utility can be found in the work involving the integration of Polyscheme's general purpose inference capabilities with specific structures designed to accommodate complex spatial reasoning tasks (Kurup & Cassimatis, 2010) (Kurup & Cassimatis, 2010). As part of this line of research, comparisons were made between versions of Polyscheme that made use of different implementations of spatial reasoning and between Polyscheme and completely domain-independent reasoning systems. As is shown in Figure 49, Polyscheme demonstrates a nearly linear performance versus problem size (green lines) when it is equipped with a dedicated spatial reasoning component (red lines). Reasoners without such a component demonstrate an enormous increase in processing time between problem instances with grid sizes between 50 and 70.

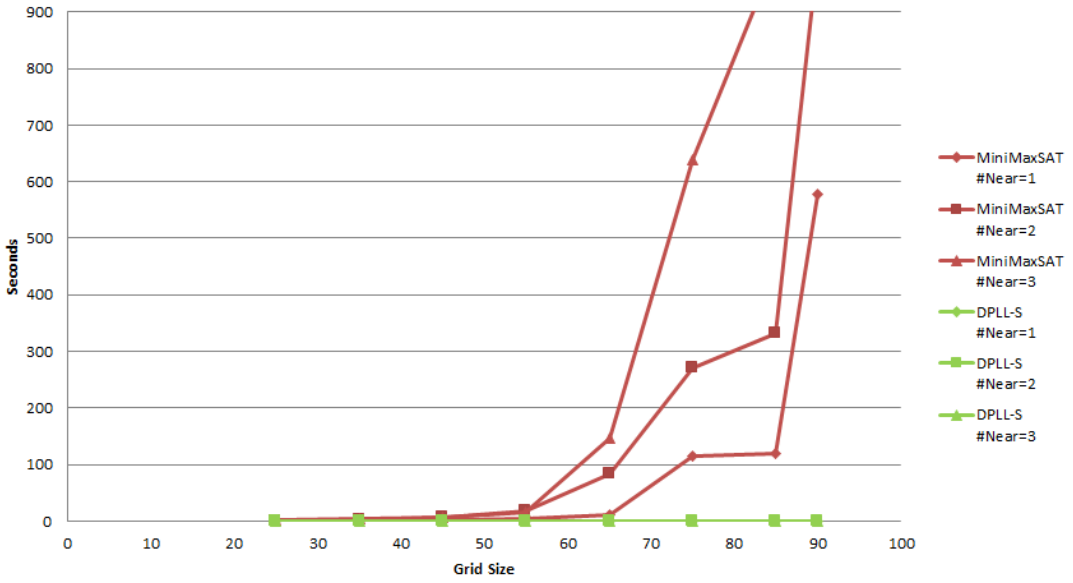

Figure 47. The performance of Polyscheme compared to domain-independent reasoners on a spatial reasoning task (Kurup & Cassimatis, 2010).

Additionally, theoretical work has been done specifically to demonstrate that Polyscheme's physical reasoning cognitive substrate hypothesis can be leveraged to create hybrid algorithms capable of performing complex NLP tasks, such as sentence parsing. An illustration of how parsing might be broken down into functions of physical reasoning is provided in Figure 50. The main kind of physical relations involved in this treatment are: 1.) $PartOf$, which designates membership in higher-level phrases, 2.) $Same$, which designates that two entities are equal in terms of the object they refer to (this is used for argument binding), 3.) $Occurs$, which designates that an event takes place during a particular time interval, and 4.) $Before$, which designates that a time interval happens before another time interval. Not shown explicitly in this diagram are two additional facets of physical reasoning: 1.) categories (e.g. types of phrases) and 2.) causation, which here takes the form of constraints that are capable of introducing new objects during inference.

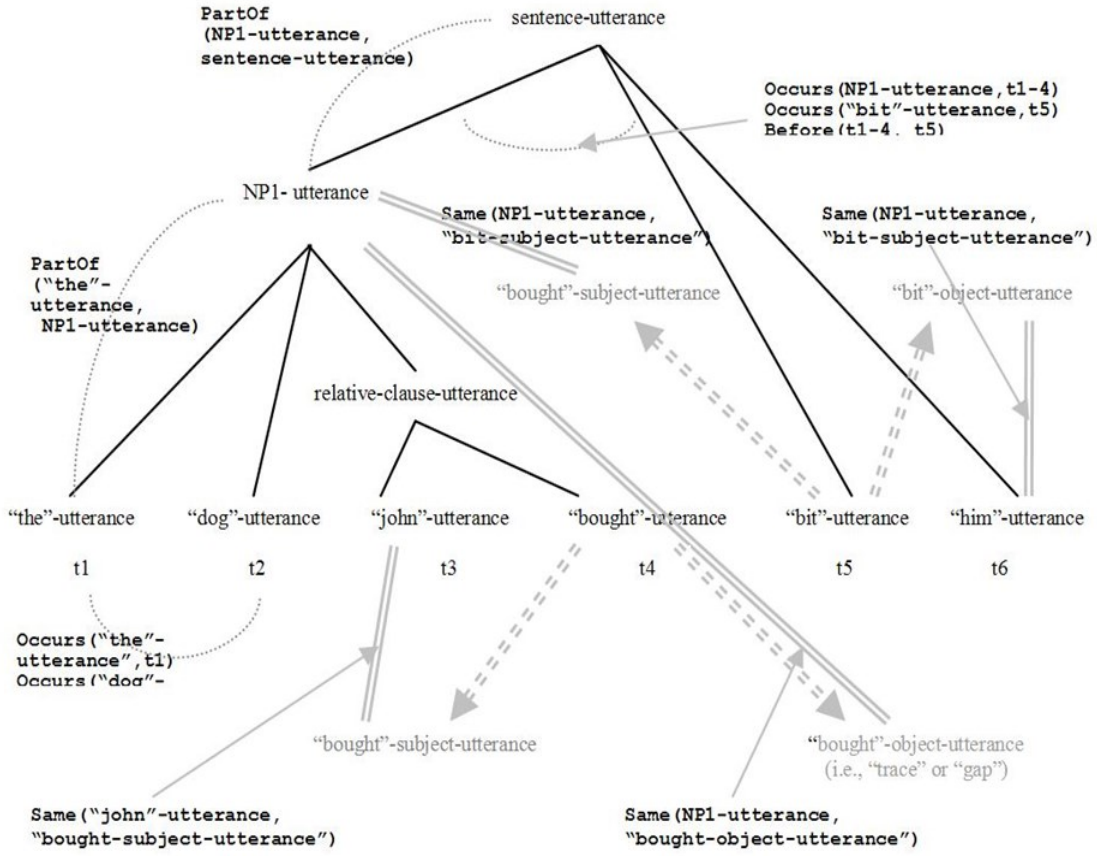

Figure 48. An illustration of how sentence parsing can be reduced to functions in Polyscheme's physical reasoning-based Human Cognitive Substrate (Cassimatis N. L., 2013).

This formulation of sentence parsing motivates the work in the following chapters of this dissertation. Inference algorithms and representations for the domains of time ($Occurs$, $Before$), object equality ($Same$), categories ($PartOf$), and causation (causal constraints) must be implemented in order to produce a parser that operates according the outline in Figure 50. Other work in Polyscheme has resulted in the creation of subsystems for reasoning about the category and causation elements (Scally, Cassimatis, & Uchida, 2011). Therefore, the bulk of the remaining work involves time and object equality. In particular, Chapter 18 will focus on mechanisms for reasoning about time and Chapter 19 will focus on mechanisms for reasoning about object equality.

18.4 Applying Cognitive Architectures to NLP System Development

Now that the deficiencies of existing approaches to human-level NLP have been discussed and the information gained from this process has led to the identification of the types of qualities that a human-level NLP system should have, the clear step forward is to attempt to implement some system that operates on these procedures. While the development of a human-level NLP system is certain to be a long process with many iterations, which might even lead to a refinement in the supposed necessary properties of such a system, it should be possible to make a first-approximation that will demonstrate whether or not there is merit to the approach towards Cognitive Architecture-drive NLP that is described above. To make this determination, it is necessary only to demonstrate that some qualitative advance is made over existing methods. Evidence of this kind of advance can be made in one of two ways.

One way would be to demonstrate that the guidelines used to create such a system were more successful than other related attempts. For instance, a project that involved the integration of world-knowledge and lower-level sentence processing has been attempted by merging the ACT-R framework with CYC (Ball, Rodgers, & Gluck, 2004). Although this sort of project is a step in the right direction, this work ran into a number of difficulties that are outlined in the paper. These difficulties ultimately stemmed from the fact that neither ACT-R nor CYC were really equipped to handle the task that was presented to them. Because CYC lacked the domain knowledge required by the project and because there was no way to leverage either CYC or ACT-R to fill these gaps, the overall result was far short of the mark that the authors had hoped to achieve.

Another way to demonstrate the relative fitness of an NLP system that is geared toward human level NLP is to compare its performance directly with the current state-of-the-art solution for some problem. An in-depth discussion of the weaknesses of major NLP-centered technology, such as Google Search or Apple's Siri Personal Assistant was provided above. It would be sufficient to conclude that a small scale system is based on better principles than those projects if it could be demonstrated to outperform one or more of them on a particular task. Since solving the general human-level NLP problem would require far more knowledge-engineering and computational resources to achieve in

the time frame of a doctoral dissertation, it is necessary that a relatively easy task be selected.

However, it is also important that such a task have certain characteristics that validate its use as a representative example. First, the task should require some amount of commonsense or world-knowledge to implement. Some of this knowledge can be hard-coded, but it should be a relatively small amount that is entirely decoupled from the interaction between that knowledge and the specific actions that are required to parse sentences (i.e. this knowledge should not contain much if any explicit information about noun phrases, agreement, etc.). Second, the task should admit some amount of dimensionality reduction to act as a proof of concept that this practice should be applicable to other domains. Again, the nature of this reduction and the mechanisms that support it should be decoupled from the specific demands of NLP algorithms. Third, it should be possible to demonstrate that it is possible to learn new information to extend the capabilities of the chosen framework within the chosen domain with a minimal amount of human intervention. As was emphasized above, one of the principal failure modes of knowledge-rich NLP systems is that they lacked any ability to capitalize on common syntactic patterns in the language being modeled. Since it has been demonstrated that knowledge-lean approaches actually can learn these types of patterns, it is necessary that this facility be included in the implementation of any human-level NLP system. Doing so will not only demonstrate that the selected approach will exhibit the requisite properties of flexibility and robustness, but also it will provide a clear direction for additional research projects along this same line.

From the above suggestions, a proposal follows as such. An adaptive cognitive architecture approach to NLP will be used to create a domain-specific NLP system that can achieve results that outstrip the current state-of-the-art offerings. As was demonstrated above, Polyscheme is the natural Cognitive Architecture to use to support this implementation. An example of a domain that has a relatively encapsulated knowledge base is that of a weather-related base, such as would be assessable via Google or Siri queries. Although it should be possible to define the task of retrieving information about temperature, precipitation, and other weather phenomena in a way that is relatively independent from knowledge of other domains, there is enough richness in the possible

queries that only a knowledge-based system would have any chance of succeeding in delivering the kind of results that people would expect. For instance, the Siri weather agent search engine is known to have exhibited a number of high profile mistakes that may be rectifiable through the use of knowledge, such as confusing place names with other place names that are similar or identical (e.g. New York, NY with New York, TX) (Golson, 2012). There are already a large number of basic weather commands that Siri can understand (see: (Nicuagain, 2011)), but does not seem to handle complicated relationships or coordination, such as *Is the weather going to be better in Binghamton or Boston tomorrow?* Since the performance bar seems to be set rather low, it does not seem to be difficult to create a weather database frontend that outperformed the Siri weather agent search engine by a large margin.

Because Polyscheme is the Cognitive Architecture that inspires the implementation of this system, it is necessary that the knowledge required to answer queries in the given domain be reducible to combinations of substrate elements. Many of the questions that arise when asking about the weather can be solved by reasoning about the passage of time and the identity of objects. Since these are all elements of Polyscheme's proposed substrate for physical reasoning, it will be possible to create a mapping between weather queries and that substrate. Thus, some of the research that will have to be done for this implementation to be successful must involve the design and incorporation of Polyscheme elements that can process time and identity in a way that can be integrated into its general framework for supporting adaptable hybrid algorithms. Once this is achieved, it should be possible to represent the required knowledge for time and identify in a way that is reliant only on Polyscheme substrate elements.

What remains after this is done is to create a model that combines such knowledge with information extracted from queries to weather databases and an empirically-driven model for natural language parsing. Recall that it has already been shown that it is possible to reduce sentence processing to Polyscheme's physical reasoning substrate. Furthermore, if enough syntactic patterns are learned by the system, then it should be straightforward to plan further research that would allow even more advanced queries. Asking questions that involve categories (e.g. kinds of weather, such as tropical storms) or quantities (e.g. lowest nightly temperature in the past 30 days)

should not differ much from the structure of questions that could be learned from a domain-independent corpus. Therefore, the proposed weather database knowledge-rich search front-end project meets all of the criteria that have been deemed necessary for the success of an NLP system that is designed to achieve human-level performance.

19. Efficient Constraint-Satisfaction in Domains with Time

Because cognition is geared towards reasoning about events in the world and since events necessarily take place over time, it is certain that time plays a major role in almost every kind of human-level inference. NLP is no exception; intricate reasoning about times impacts NLP tasks in a large number of ways. On one hand, at a high level, language is used to communicate information about events, including when those events occurred both from an absolute perspective and from a relative perspective of other events known to the parties privy to such communication. To handle this requirement, many languages, such as English, utilize a complex tense system, as well as allowing the specification between instantaneous and habitual events and allowing the specification of durative events using the progressive aspect (Kearns K. , 2000). On the other hand, language itself can be considered to consist of events, which may be ordered in time differently than the order in which they conceptually occur. For instance, it is possible to start a sentence with a clause that acts as an argument that would normally come after a verb later in a sentence. Consider that the sentence *He gave a toy to the child* can be rearranged to read *To the child, he gave a toy*. Even though *to the child* appears before *give*, it still must be considered an argument of *give*. Reasoning about time aids in solving these more implicit ordering problems that occur during language processing.

In order to incorporate reasoning about time into an architecture that is general enough to also reason about language, it is necessary to encode information in the most domain-neutral way possible. One of the best candidate methods for this is to cast these reasoning problems as Satisfiability (SAT) problems. Satisfiability (SAT) testing has been used effectively in many inference, planning and constraint satisfaction tasks. However, since SAT constraints are defined over atomic propositions, domains with state variables that change over time can lead to extremely large search spaces. This poses both memory- and time-efficiency problems for existing SAT algorithms. In this paper, we propose to address these problems by introducing a language that encodes the temporal intervals over which relations occur and an integrated system that satisfies

---

Portions of this chapter previously appeared as: Bignoli, P. G., Cassimatis, N. L., & Murugesan, A. (2010). Efficient Constraint-Satisfaction in Domains with Time. Proc. of the 3rd Conf. on Artificial General Intel. (AGI-2010). Lugano, Switzerland.

constraints formulated in this language. Temporal intervals are presented as a compressed method of encoding time that results in significantly smaller search spaces. However, intervals cannot be used efficiently without significant modifications to traditional SAT algorithms.

Using the Polyscheme cognitive architecture, we created a system that integrates a DPLL-like SAT-solving algorithm with a rule matcher in order to support intervals by allowing new constraints and objects to be lazily instantiated throughout inference. Our system also includes constraint graphs to compactly store information about temporal and identity relationships between objects. In addition, a memory retrieval subsystem was utilized to guide inference towards minimal models in common sense reasoning problems involving time and change. We performed two sets of evaluations to isolate the contributions of the system's individual components. These tests demonstrate that both the ability to add new objects during inference and the use of smart memory retrieval result in a significant increase in performance over pure satisfiability algorithms alone and offer solutions to some problems on a larger scale than what was possible before.

19.1 Introduction and Literature Review

SAT has emerged as a popular technique for automated reasoning because it is efficient and domain general (Russell & Norvig, 1995). Furthermore, because SAT-solvers are declarative, the same algorithm can be used for different aspects of intelligence (i.e., inference, explanation, etc.). It is also possible to create SAT procedures that are guaranteed to be complete over finite search spaces. If optimal solutions are required, instances of general SAT can be easily adapted to weighted SAT. For these reasons, many important AI applications have been successfully framed as SAT problems: planning (Kautz & Selman, 1999), computer-aided design (Marques-Silva & Sakallah, 2000), diagnosis (Smith & Veneris, 2005), and scheduling (Feldman & Golumbic, 1990). Although SAT-solvers have successfully handled problems with millions of clauses, tasks that require an explicit representation of time can exceed their capacities.

Adding a temporal dimension to a problem space has the potential to greatly expand search space sizes because SAT algorithms propositionalize relational constraints.

The most direct way to incorporate time is to have a copy of each state variable for every time point over which the system reasons. This increases the number of propositions by a factor equal to the number of time points involved. Since SAT algorithms generally become slower as the size of a problem increases, adding a temporal dimension to even relatively simple problems can make them intractable.

Although problems with time require more space to encode, the true expense of introducing time stems from the additional cost required to find a SAT solution. Consider a task that requires the comparison of all the possible ways that a car can visit three locations in order. If the problem has no reference to time points, there is only one solution: the car just moves from *a* to *b* to *c*. On the other hand, there are clearly more possibilities, since the car could potentially move or not move at every time. Compared to other SAT-solvers, LazySAT (Singla & Domingos, 2006), which lazily instantiates constraints, is less affected by the increased memory demands of larger search spaces. Unfortunately, lazy instantiation will not increase the tractability of larger problems with respect to runtime.

There is, however, a more efficient way of representing time. Since it is unlikely that the truth value of a proposition will change at every time point, temporal intervals can be used to denote segments of contiguous time points over which its value is constant. This practice alleviates the need to duplicate all propositions at every time point for most problem instances, thus significantly reducing the search space size. Intervals also mitigate the arbitrary granularity of time because they are continuous and scale independent.

However, existing SAT solvers cannot process intervals efficiently because they do not allow new objects to be introduced during the course of inference. It is clearly impossible to know exactly which temporal intervals will be required. Therefore, every possible interval must be defined in advance. Since $^{(n-1)(n-2)}/_{2}$ unique intervals can be defined over n times, there would be little advantage to use them with current searching methods.

To capture the benefits of SAT while supporting the use of intervals, we created an integrated system that combines a DPLL-like search with several specialized forms of inference: a rule matcher, constraint graphs, and memory retrieval. Rule matching allows

our system to both lazily instantiate constraints and introduce new objects during inference. Constraint graphs compactly store temporal and identity relationships between objects. Memory retrieval supports common sense reasoning about time. Although SAT has previously been applied to planning (Shanahan & Witkowski, 2004) and reasoning (Mueller, 2004) with the event calculus, we present a novel approach.

19.2 Definition of the New Logic Language

We have specified a language that can express relationships between objects at different points in time. This language incorporates temporal intervals and a mechanism for introducing new objects during inference. For example, in path planning tasks, it is often useful to have a condition such as: If at location *p1* and must be at location *p2*, which is not adjacent to *p1*, then move to some location *px* that is adjacent to *p1* at the next possible time. We write this constraint as:

$$\begin{aligned} &Location(?x,?p1,?t1) \wedge Location(?x,?p2,?t2) \wedge \neg Same(?p1,?p2,E) \Rightarrow (\infty) \\ &Adjacent(?p1,?px,E) \wedge Meets(?t1,?tx,E) \end{aligned}$$

Equation 1: An Example of a Constraint with Time

Note that we prefix arguments with a '?' to denote variables. Variables permit this one constraint to apply to all locations in the system. Additionally, we can assign weights to constraints as a measure of their importance. Because this constraint holds for any path, it is given an infinite weight to indicate that it must be satisfied.

Formally, constraints have the form:

$$A_1 \wedge \ldots \wedge A_m \Rightarrow (w)\, C_1 \wedge \ldots \wedge C_n$$

Equation 2: The Basic Form of a Polyscheme Constraint.

where w is a positive rational number, m ≥ 0 and n ≥ 1. A_i and C_j are first order literals. Literals have the form $P(arg_1, \ldots, arg_n)$, where P is a predicate, arg_i is a term, and arg_n must be a time point or interval. Terms that are prefixed with a '?' are variables; others are constant "objects." A grounded predicate is one in which no term is a variable. Predicates specify relations over the first n-1 terms, which hold at time arg_n. There is a special type of relation called an attribute. Attributes are predicates, P, with three terms, o, v, and t, such that only one relation of the form $P(o, v, t)$ holds for each o at every t.

The negation of a literal is expressed with the ¬ operator. Every literal is mapped to a truth value.

If all of the literals in a constraint are grounded, then the constraint itself is grounded. Only grounded constraints can be satisfied or broken, according to the truth value of its component literals. A constraint is broken iff every antecedent literal is assigned to true and every consequent literal is assigned to false. The cost of breaking a constraint is given by w, which is infinite if the constraint is hard.

Some predicates and objects are included in the language. For instance, Meets, Before, and Includes are temporal relations that are similar to the predicates in Allen's interval calculus (Allen, 1983). We reserve a set of objects of the form {*t1*, …, *tn}*, where n is the number of times in the system. This set is known as the native times. Another time, E, denotes eternity and is used in literals whose assignments do not change over time.

A model of a theory in this language consists of a complete assignment, which is a mapping of every literal to a truth value. Valid models are those such that their assignment permits all hard constraints to be satisfied. All models have an associated cost equal to the cost of its broken constraints. Each theory has many valid models, but it is often useful to find one of the models with the minimum cost. For instance, this process can perform change minimization, a form of commonsense reasoning motivated by the frame problem (Shanahan, 1997).

19.3 System Architecture

We created an integrated system using the Polyscheme cognitive architecture (Cassimatis & L., 2001) in order to efficiently solve problems with time. This approach allowed us to glean the benefits of SAT while capitalizing on the properties of specialized forms of inference. It is easiest to describe how our system works by framing it as a DPLL-like search. DPLL (Davis, Logemann, & Loveland, 1962) performs a branch-and-bound depth first search that is guaranteed to be complete for finite search spaces. The algorithm searches for the best assignment by making assumptions about the literals that appear in its constraint set. An assumption consists of selecting an unassigned literal, setting it to true or false, and then performing inference based on this assignment. When

necessary, the search backtracks to previous decision points to explore the ramifications of making the opposite assumption.

The DPLL-FIRST procedure in Algorithm 1 (depicted in Figure 51) takes a set of constraints, *c*, as input and outputs an assignment of literals to truth values that minimizes the cost of broken constraints. In the input constraint set, there must be at least one fully grounded constraint with a single literal. Such constraints are called facts. Within the DPLL-FIRST procedure, several data structures are declared and passed to DPLL-RECUR, which is illustrated in Algorithm 2 (depicted in Figure 52). First, there is a structure, *assign*, which stores the current assignment of literals to truth values. The facts specified in the input are stored in *assign* with an assignment that corresponds to the valence of the fact's literal. Second, there is a queue, *q*, which stores the literals that have been deemed relevant by the system in order to perform inference after the previous assumption. At the beginning, *q* contains the facts in the input. Third, *b* stores the best total assignment that has been found so far. Fourth, the cost of *b* is stored in *o*. Initially, *b* is empty and *o* is infinite.

Algorithm 1. DPLL-FIRST(*c*):

    return DPLL-RECUR(*c*, *assign*, *q*, *o*, *b*)

Figure 49. The DPLL-First algorithm.

Each time DPLL_RECUR is called, it performs an elaboration step that infers new assignments based on the current assumption. Initially, when there is no assumption, the elaboration step attempts to infer new information from the constraints specified in the input. After elaboration, the current assignment is examined to determine if one of the three termination conditions is met. The first condition is if the assignment is contradictory because the same literal has been assigned to both true and false. The second condition is if the cost of the current assignment exceeds the lowest cost of a complete assignment that has been discovered in previous iterations. Since new assignments can never reduce the total cost, it is unnecessary to continue searching. The third condition is if the current assignment is complete. In all of these cases, the search backtracks to a previous assumption and investigates any remaining unexplored

possibilities. Afterwards, the search selects an unassigned literal and creates two new branches in the search tree: one where the literal is assumed to be true and one where it is assumed to be false. DPLL-RECUR is then invoked on those subtrees and the assignment from the branch with the lower cost is returned.

Algorithm 2. DPLL-RECUR(*c*, *assign*, *q*, *o*, *b*)

```
call ELABORATION(c, assign, q)
if Contradictory(assign) then
        return Fail
else if Cost(assign) > o
        return Fail
else if Complete(assign)
        return assign
end if
u ← next element of q
newassign ← assign with u assigned to true
b1 ← call DPLL-RECUR(c, newassign, o, b)
newassign ← assign with u assigned to false
b2 ← call DPLL-RECUR(c, newassign, o, b)
if Cost(b1) < Cost(b2) then
        return b1
else
        return b2
end if
```

Figure 50. The DPLL-RECUR algorithm.

The elaboration step in basic DPLL is called unit-propagation. Unit-propagation examines the current assignment to determine if there are any constraints that have exactly one literal unassigned. If such constraints exist and exactly one assignment (i.e., true or false) for that literal satisfies the constraint, then DPLL makes that assignment immediately instead of through a later assumption. Our system augments this basic technique by introducing several more specialized forms of inference.

To understand the importance of elaboration, consider that all of the best available complete SAT-solvers are based on some version of DPLL (Moskewicz, Madigan, Zhao, Zhang, & Malik, 2001) and (Een & Sorensson, 2005). DPLL is so effective because its elaboration step eliminates the need to explore large numbers of unnecessary assumptions. It is more efficient to infer assignments directly rather than to make assumptions, because each assumption is equivalent to creating a new branch in DPLL's

abstract search tree. Elaboration also allows early detection of contradictions in the current assignment.

Despite its elaboration step, DPLL is unable to handle the large search spaces that occur when time is explicitly represented. The goal of our approach is to improve elaboration by using a combination of specialized inference routines. Previous work (Cassimatis et al., 2010) has outlined the implementation of SAT solvers in Polyscheme. Following that approach, we implemented DPLL using Polyscheme's focus of attention. One call to DPLL-RECUR is implemented by one focus of attention in Polyscheme. Logical worlds are used to manage DPLL assumptions. For each assumption, an alternative world is created in which the literal in question is either true or false. Once Polyscheme focuses on an assumption literal, it is elaborated by polling the opinions of several specialists. These specialists implement the specialized inference routines upon which our system relies. One of these specialists, the rule matcher, lazily instantiates grounded constraints that involve the current assumption. The assignment of a literal is given by Polyscheme's final consensus on the corresponding proposition. This elaboration constitutes the main difference between our system and standard DPLL.

Our elaboration step, which is illustrated in Algorithm 3, Elaboration, (depicted in Figure 53), loops over the literals that have been added to the queue because their assignments were modified by previous inference. Two procedures are performed on each of these literals. First, a rule matcher is used to lazily instantiate grounded constraints from relevant variable constraints provided in the input. Relevant constraints are those that contain a term that corresponds to the current literal in focus. These constraints are "fired" to propagate truth value from the antecedent terms to the consequent terms. Newly grounded literals, which may contain new objects, are introduced during this process. Any such literals are added to the assignment store and the queue.

The second procedure involves formulating an assignment for the current proposition based on suggestions from the various components of the system. For instance, the temporal constraint graph is queried here in the case that the proposition being examined describes a temporal relationship. Likewise, the identity constraint graph would be queried if the examined proposition was an identity relationship. In the

extended system, this is the step at which the memory retrieval mechanism would be utilized. These opinions are combined with the old assignment of the proposition to produce a new assignment. If the new assignment differs from the old one, the literal is placed back on the queue.

Algorithm 3. ELABORATION(*c*, *assign*, *q*):

```
while q is not empty do
        l ← the next element in q
        ris ← call Match(l, c, q)
        delta ← ∅
        for each ri in ris do
                delta ← delta ∪ call Propagate(ri, assign)
        end for
        rs ← the rule system's opinion on l
        tc ← the temporal constraint graph's opinion on l
        ic ← the identity constraint graph's opinion on l
        mr ← the memory retrieval system's opinion on l
        c ← call Combine(rs, tc, ic, mr)
        if c ≠ l's assignment in assign then
                delta ← delta ∪ l
        end if
        q ← q ∪ delta
end while
return
```

Figure 51. The Elaboration algorithm.

19.4 Rule Matching Component

Lazy instantiation is efficient because it avoids the creation of constraints that do not need to be considered to produce an acceptable assignment. We accomplish lazy instantiation by treating constraints with open variables as templates that can be passed to a rule matcher. The rule matching component (RMC) attempts to bind the arguments of the last dequeued literal to variables in the constraint rules. A binding is valid if it allows

all variables in the antecedent terms of a constraint to be bound to objects in the arguments of literals that are stored in the system's memory. All valid bindings are evaluated as constraints by propagating truth value from the antecedents to the consequents. A constraint is considered broken only if the grounded literals in its antecedent terms have been assigned to true and the propositions in its consequent terms have been assigned to false.

A simple example will illustrate the binding process. Let the literal currently being assigned be represented by the following predicate: $Location(car1, road, t1)$. If $Location(?x, ?y, ?t1) \Rightarrow (10)Location(?x, ?y, ?t2)\text{^}Meets(?t1, ?t2, E)$ appears as a constraint, then the following fully grounded instance is created:

$$Location(car1, road, t1) \Rightarrow (10)Location(car1, road, tnew)\text{^}Meets(t1, tnew, E).$$

Equation 3: An Example of a Fully-Grounded Polycheme Constraint.

In this case, *?x* binds to *car1*, *?y* binds to *road*, *?t1* binds to *t1*, and *?t2* binds to a new object, *tnew*. If *Location(car1, road, t1)* is assigned to true and *Location(car1, road, t_new)* is later assigned to false, then any models that contain that assignment will accrue a cost of 10.

19.5 Temporal Constraint Graph Component

As the number of objects in a problem instance increase, so do the number of literals and constraints. For instance, when time objects are introduced, it is often necessary to know how those times are ordered. If there are n times in the system, approximately $n^2/2$ literals are required to represent all of the values of a binary relation over those times. Usually, only a small portion of these literals provide useful information for solving the problem. Instead of eagerly encoding all temporal relations, we created a component that could be queried on demand to determine if a given relation holds according to the current knowledge of the system. This component represents relations in graphical form.

Using a graph enables the system to derive new entailed relations without storing them explicitly as propositions. All of the fundamental temporal relationships described

by Allen can be represented in the following way. Whenever a literal that involves such a relationship is encountered, the temporal constraint graph (TCGC) decomposes the time object arguments into three parts: the start point, the midpoints, and the end point. These parts form the nodes of the graph. Edges are created between two nodes in the graph if their temporal relationship is known.

Every interval relationship can be derived from only two types of relationships on the parts of times: Before and Equals. A time object's start point is defined to be before its midpoints and its midpoints are before its end point. By creating edges between the parts of different time objects, it is possible to record relationships between the objects themselves. For instance, to encode *Meets(t1, t2, E)* in the graph, one would use the relationship: *Equals(end-t1, start-t2, E)*. To illustrate why the graph is an efficient way to store this information, consider the following example. If it is known that *Meets(t1, t2, E)* and *Meets(t2, t3, E)*, then the graph can be traversed to find *Before(t1, t3, E)*, among other relationships. Thus, these propositions are stored implicitly and do not need to be assigned unless they are present in a grounded constraint.

19.6 Identity Constraint Graph Component

The identity constraint graph component (ICGC) is similar to the TCGC, but it handles propositions about the identity of objects. This graph consists of nodes that represent objects and edges that represent either equality or inequality. By traversing the graph, it is possible to capture the transitivity of the identity relation. Although a rule could be used to generate the transitivity property, doing so has the potential to drastically increase the number of propositions over which DPLL must search.

Another important use for the ICGC is that it can detect inconsistencies in truth assignments to identity propositions. Consider that the following set of identity propositions is known: *Same(a, b)*, *Same(b, c)*. Then, the proposition, ¬*Same(a, c)*, is examined and assumed to be true. Clearly this is inconsistent, but without including a rule that defines the properties of identity, the system will continue to perform inference based on these facts until an explicit contradiction is encountered. Traversing the edges connecting *a*, *b*, and *c* in the graph will indicate that this scenario is contradictory. This information can be reported to the system as a whole during elaboration.

19.7 Memory Retrieval Component

Systems that explicitly represent time often also have to reason about change. However, in situations with imperfect knowledge, reasoning about change can be difficult. Consider the following information about the color of a car, assuming that Location is an attribute: *Color(car, red, t1)*, *Color(car, blue, t6)*, *Color(car, green, t11)*. In between *t1* and *t6* and *t6* and *t11*, it is consistent for the car to be any color. However, in many scenarios, the car would remain *red* until it was painted *blue* and then *blue* until it was painted *green*. Although the world is dynamic, it also exhibits inertia. The principle of change minimization states that changes to particular objects are relatively infrequent and when changes do occur, they will be salient. It is worthwhile for a reasoning system to bet on such recurring patterns, because doing so significantly reduces the complexity of many problems.

We can frame the change minimization problem as a weighted SAT problem as follows. Given a set of attribute literals that involve the same first term, but whose second term changes over time, determine the minimum number of changes required to explain the data. To this end, constraints can be defined so that the least costly models will be those that exhibit the smallest possible number of attribute value changes. The memory retrieval component (MRC) is designed to accommodate this procedure by regulating which attribute values appear in grounded constraints. Only attribute values with some prior evidence in memory are considered. As an example, the car could have been *purple* at *t2* and *brown* at *t3*, but models that contain such literals are automatically excluded.

To control which values are considered, the MRC makes a copy of each literal that encodes an attribute with a native time index. This copy is modified to contain a time index that corresponds to an open-ended interval. Only when new attribute values are observed will that interval's endpoints be constrained. For instance, when the system elaborates *Color(car, red, t1)*, a new literal *Color(car, red, t-car-red)* will be introduced. Because it uses a content-addressable retrieval system, the MRC will henceforth report that the color of the car is likely to be red at every time point until new information is discovered. If the literal *Color(car, blue, t6)* is observed, then the literal *Color(car, blue, t-car-blue)* will be introduced. Also, constraints will be added to limit the right side of the *t-car-red* interval at *t6* and the left side of the *t-car-blue* interval at *t1*.

Even with this optimization, change minimization is expensive if performed naively, since there are $^{(n-1)(n-2)}/_{2}$ possible transitions between n attribute values if nothing is known about when the values hold relative to each other. For instance, if a car is seen to be *red*, then *blue*, then *green*, the naïve formulation will consider such possibilities as a change from *green* to *red* even though this is clearly impossible. The change detection mechanism in the MRC utilizes the TCGC in conjunction with content-accessible memory to significantly reduce the number of impossible changes that are considered by the system. When an attribute literal containing an interval is elaborated, the only values that are adjacent in time to the current interval will be considered. Since the intervals do not have completely fixed endpoints, these neighboring times can be detected by looking at what times the current interval could possibly include. For instance, once it has been established that there is a change between *t1* and *t6*, the interval created around *t1* can no longer include *t11*, so the location *t1* will not appear as a possible previous state of the location at *t11*.

19.8 Results

Our system was designed to improve the efficiency of applying SAT-solving to problems that involve an explicit representation of time. In order to accomplish this goal, we used Polyscheme to implement a DPLL-like algorithm with specialized forms of inference that permitted the creation of new objects. The ability to introduce new objects allowed us to use intervals to reduce the search space of these problems.

In order to test the level of improvement gained by the ability to add new objects independently of other techniques, we ran an evaluation with the memory retrieval component deactivated. The task we selected was optimal path planning through space. We represented this space as a graph. Although the particular problem we used did not model a changing environment, an explicit represent of time would be required if actions had side effects or if objects in the environment had changing states. For instance, a particular environment might contain walls that crumble over time. We asked the system to find the shortest valid path between two particular locations on the graph. Valid paths consisted of movements between adjacent locations that did not contain active obstacles.

These types of problems are important to the object tracking and motion planning domains.

We initially compared our system against LazySAT because it is similar to our approach in that it also supports the lazy instantiation of constraints. Through experimentation, we determined what values of LazySAT's parameters enabled high performance on this task. LazySAT, however, is based on a randomized local search, which is inefficient for many structured domains. Therefore, we ran the same set of tests on MiniMaxSAT (Heras, Larrosa, & Oliveras, 2007), which is one of the best DPLL-based systematic SAT-solvers. Because Markov logic is not complete and cannot report unsatisfiability, we were forced to select a configuration of space in which a valid path existed.

The evaluation problem involved a 9-location graph with one obstacle, which had to be circumvented. We were limited to 9 locations because the performance of the systems degraded on larger problems. To determine how well each system handled time, we created ten versions of the problem from 5 to 50 time points in increments of 5. For each condition and each system, we ran 10 trials and recorded the average runtime. These results are displayed in Figure 54. While our system was consistently better than LazySAT, MiniMaxSAT outperformed both until it had to contend with more than 30 time points. Our system required approximately constant time to solve the problem due to the fact that intervals make this task equivalent to the case of planning without explicit time points.

A second evaluation was conducted to show that the memory retrieval subsystem allowed change minimization problems to be solved efficiently. We ran these tests on the system with and without the MRC activated. These problems consisted of a number of attribute value observations that involved the color of an object. For instance, suppose the following facts were given as input: *Color(car, red, t1)*, *Color(car, blue, t6)*, and *Color(car, green, t11)*. The system would then be tasked with finding the least costly model that explained this data, namely that the color changed from *red* to *blue* and then from *blue* to *green*.

The results from this evaluation are depicted in Figure 55. Not only does enabling the MRC permit the change minimization problems to be solved in less time than with

the basic system, but it also increases the upper limit on problem size. Without the MRC, our system ran for over 50,000 seconds attempting to solve the 5 attribute value problem. These results demonstrate that the elimination of irrelevant constraints is a powerful technique for improving the performance of SAT-solvers.

19.9 Conclusion

Although SAT is typically an efficient approach to domain-general problem solving, it does not scale well to the large search spaces that result from tasks that require an explicit representation of time. Temporal intervals help to reduce the size of such problems, but can be used effectively only with SAT-solvers that permit the introduction of novel objects throughout inference. Hence, we created a system that combines specialized inference techniques with a DPLL-like algorithm. We employed a rule matcher in that system to support the creation of new objects. Additionally, we reduced search spaces with constraint graphs that compactly represent temporal and identity relationships. This system was shown to outperform MiniMaxSAT and LazySAT in a series of evaluations involving a simple path planning domain.

When time is represented explicitly, it is also beneficial to incorporate common sense reasoning that exploits common patterns in real-world problem instances. Towards this end, a memory retrieval subsystem was developed that prevented the exploration of fruitless paths in the DPLL search tree under certain conditions. This technique was demonstrated to increase the efficiency of how our system solves the change minimization problem.

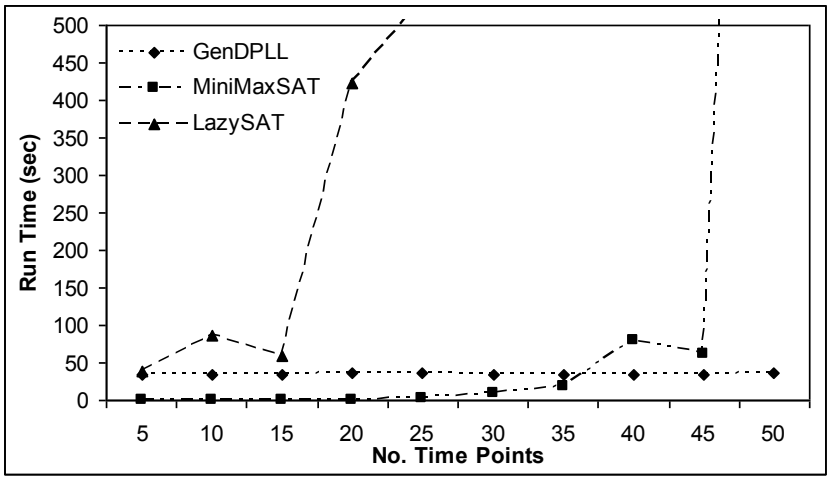

Figure 52. Path planning problem results.

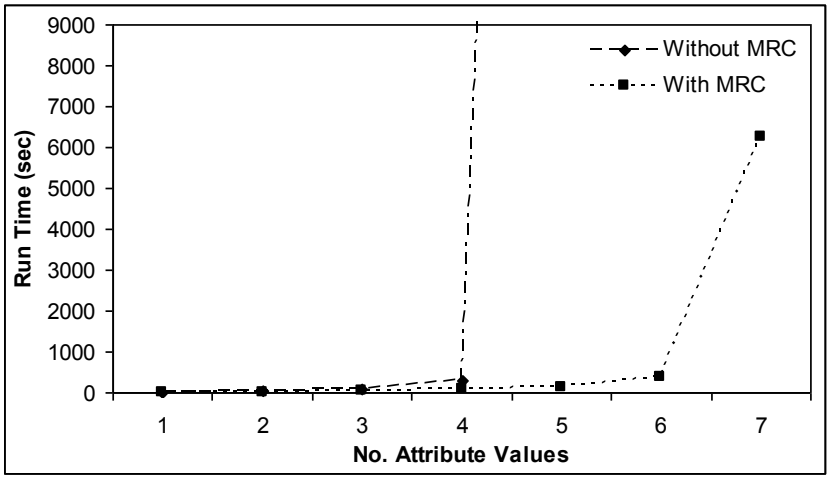

Figure 53. Change minimization results.

20. Inference in Relational Domains with Equality

Reasoning about equality is similar to reasoning about time in that it is a major facet both of human cognition in general and of NLP in particular. Whereas the temporal aspects of communication are more concerned with events, the aspects of communication that deal with equality are more concerned with entities (although note that events can be reified as entities, in which case, equality is also concerned with events). A prime example of this type of reasoning plays a major role in the task of reference resolution. Reference resolution essentially is a task of assigning mentions of entities into cliques of equal objects. There is also a more fundamental application for reasoning about equality while processing language. During discourse, there are many structures, such as transitive verbs and ditransitive verbs, which take arguments. One of the prime tasks of parsing a sentence is to determine which other structures serve as those arguments. This process can be cast as a problem of determining equality cliques, as well.

Aside from concerns that arise in NLP problems, many important real-world problems involve reasoning about the equality of entities. Although relational weighted satisfiability approaches have been effective in solving many problems, reasoning with equalities can cause difficulties for these methods. This is due to the fact that the equality of entities can lead to the equality of clauses, which is not accounted for by the traditional cost function for models of these theories. The equality of clauses leads to non-monotonic changes in cost during systematic search and violates the assumptions of widely used branch-and-bound heuristics.

To address these problems, we define a new language with equality and introduce a new cost function for models in that language. We next describe a lazy algorithm based on DPLL for finding models in that language. Finally, we introduce a new search heuristic for this algorithm based on equality relationships and describe experiments that demonstrate how it improves performance.

20.1 Introduction

The versatility and efficiency of framing inference as a satisfiability problem have been demonstrated through a large body of AI-related work. Problems that have been translated into relational SAT theories include: planning (Kautz & Selman, 1996),

theorem proving (Kim & Zhang, 1994), and probabilistic inference (Domingos, Kok, Poon, Richardson, & Singla, 2006). On the other hand, reasoning about object equality is required for many AI applications: path planning (Divelbiss & Wen, 1997), natural language understanding (Cassimatis N. L., 2004), entity resolution (Singla & Domingos, 2006), etc.

A successful approach for dealing with weighted relational theories is to translate them into MaxSAT problems (Singla & Domingos, 2006). However, this method has difficulties with some problems that involve object equality due to the way that the costs of models are traditionally calculated. Consider theory T: $\{P(a) \Longrightarrow Q(a)[0.25]$, $P(b) \Longrightarrow Q(a)[0.25]$, $a = b[1]\}$. Let $M1$ be a partial model of T where the first two clauses are not satisfied, but $a = b$ is unresolved. Let $M2$ be a model of T that is the same as $M1$ except that $a = b$ is true. Existing formulations of MaxSAT calculate the total cost of each of $M1$ and $M2$ to be 0.5. We claim that if $a = b$ is true, then $P(a) \Longrightarrow Q(a)[0.25]$ and $P(b) \Longrightarrow Q(a)[0.25]$ are equivalent and should not independently contribute to the model cost. Therefore, the cost of $M2$ intuitively should be 0.25. Not accounting for the equality of clauses[66] can thus result in incorrect inferences.

For example, consider the following scenario involving a version of Russian roulette with blanks. Each time the player pulls the trigger and produces a gunshot, there is more evidence that he is dead. Some gunshots might be the same sound (i.e. echoes). The task is to decide if the player is dead or alive. Using a naïve encoding (which will be shown below) of this problem, existing MaxSAT-based approaches would ignore the fact that echoed gunshots should not increase the evidence that the player is dead.[67]

The fact that equality relations can reduce the cost of models during the course of inference is incompatible with the branch-and-bound heuristic (Land & Doig, 1960) for systematic MaxSAT algorithms. This heuristic requires that model costs monotonically increase as more literals are assigned truth values. Branch-and-bound greatly increases efficiency, so it is desirable to identify alternative heuristics.

---

[66] Note that we may use *clause* to refer to both logical formulae in a theory and the traditional concept of a MaxSAT clause.

[67] There are encodings that circumvent this issue, but they require appending all possible truth assignments of all possible equality relations to the end of every theory clause. This acts as a pre-computed look up table for the correct weights. Such tables have 2^{n^2} entries, where n is the number of objects in the theory.

To address these problems, we introduce a new relational language that accommodates object equality. We demonstrate how theories in this language can be translated to instances of MaxSAT. Also, a new cost function over those instances is presented that correctly and efficiently takes object equality into account. A lazy instantiation MaxSAT algorithm with this cost function is provided. Finally, a variable-ordering heuristic is formulated and empirically evaluated. It illustrates the potential for heuristics based specifically on equality relations to speed up inference and mitigate the invalidation of branch-and-bound heuristics.

20.2 Language

We propose a language for expressing weighted relational theories. The basic component of our language is a term, which is a name that refers to an object in some problem domain. For instance, terms can refer to physical entities (e.g. $player1$), reified events (e.g. $soundEv1$), categories (e.g. $shot$), etc. Terms with identical names must refer to the same underlying object. However, there is no uniqueness of names in our language, so terms with different names might refer to the same object.

Our language uses relational literals to express how terms are related. An example literal is: $EchoOf(soundEv2,\ soundEv1)$. This literal states that $soundEv2$ is an echo of $soundEv1$. To handle the lack of uniqueness of names, we introduce an equality predicate, =. We encode that two terms with different names, a and b, refer to the same entity as: $a = b$. Literals can either be true or false. A special literal, $True()$, is always true.

We use clauses to represent constraints on the values of literals. Elaborating on the Russian roulette example above, we encode some of its clauses in theory T_{RR}: $\{EchoOf(soundEv2,\ soundEv1)[\infty]^a$, $Hear(soundEvt1)[\infty]^b$, $Hear(soundEvt2)[\infty]^c$, $IsA(soundEv1, shot)[\infty]^d$, $IsA(soundEv2, shot)[\infty]^e$, $Player(?a)[\infty]^f$, $\neg Dead(player1)[3]^g$, $EchoOf(?snd1,\ ?snd2) \Rightarrow ?snd1 = ?snd2[\infty]^h$, $Hear(?s) \wedge IsA(?s,\ shot) \wedge Player(?a) \Rightarrow Dead(?a)[2]^i\}$. These clauses can be interpreted in the following way: [a-f] the literals in the clauses are true, [g] there is a cost of 3 if $player1$ is dead, [h] for all objects, if one object is the echo of another object, then those objects must be equal, and [i] for all objects, there is a cost of 2

if a shot is heard and an agent is playing Russian roulette, but the agent is not dead. In general, clauses have the form of logical conditionals over literals in which variables are implicitly universally quantified. Note that literals can be negated in clauses.

Formally, a theory, T, consists of a set of clauses. Clauses have the form: $A_1 \wedge \cdots \wedge A_m \Rightarrow C_1 \wedge \cdots \wedge C_n[w]$, where A_i and C_i are possibly negated literals; $m \geq 0$; $n \geq 1$; w is a positive number or infinity. A_i are antecedents. C_i are consequents. Clauses where $m = 0$ and $n = 1$ are called facts. We can abbreviate the fact $\Rightarrow C_1[w]$ as $C_1[w]$. Clauses where $w = \infty$ are called hard. Literals have the form: $P(arg_1, \cdots, arg_n)$, where P is a predicate; arg_i is an object name; and $n \geq 0$. Names beginning with ? are variables.

20.3 Translation and New Cost Function

Our overall goal is to produce an algorithm that properly takes equality into account when calculating the cost of a model. It is easiest to describe our approach using the language defined above, which requires us both to define the notion of a model for a theory and to demonstrate that there is a translation of theories to MaxSAT instances. We proceed by defining a translation, $Trans$, from a theory, T, to a MaxSAT instance, S: $S = Trans(T) = CNF(Ground(T + EA))$. Next we show how the original MaxSAT model cost function fails to account for object equalities. Finally, we introduce a new cost function on models that properly deals with object equalities.

20.3.1 Introduction of Equality Axioms

Recall that we have defined a general equality predicate, $=$, in the Language section. There are several commonly recognized properties of equality. For instance, the equality relation is reflexive, symmetric, and transitive. Also, if two objects are equal, then they should have the same attributes.[68] We want to ensure that our theories are consistent with these properties of equality. Thus, we append the following theory, EA, to every theory, T: $\{?x = ?x \Rightarrow \neg True()[\infty],\ ?x = ?y \Rightarrow ?y = ?x[\infty],\ ?x = ?y \wedge ?y =$

[68] This follows from Leibniz's Law of the Identity of Indiscernibles, which is introduced in (Leibniz & Alexander, 1998).

$?z \Rightarrow ?x = ?z[\infty]$, $?x = ?y \wedge ?p(?x, ?v) \Rightarrow \wedge ?p(?y, ?v)[\infty]\}$. These clauses encode the reflexivity, symmetry, transitivity, and identical attributes properties, respectively.

20.3.2 Translation to Grounded Form

To help define the notion of a model for a theory in our language, we will now demonstrate how any theory, T, can be translated into a set of fully grounded clauses. A clause is said to be grounded if all of its literals are grounded. A literal is said to be grounded if it contains no variables. A grounded literal is assigned a truth value of either true or false. Consider the following theory, $T1$: $\{P(a,b)[\infty], P(?x, ?y) \Rightarrow Q(?x, ?y)[\infty]\}$. We can ground $T1$ by setting the variables that occur in the second clause with the objects that appear in the first. In this case, a and b are assigned to $?x$ and $?y$, producing: $P(a,b) \Rightarrow Q(a,b)[\infty]$ and $P(b,a) \Rightarrow Q(b,a)[\infty]$.

Formally, we define a function $Ground(T)$ that translates a theory, T, into its grounded form, G: $G = Ground(T) = \bigcup_{c \in C} ClauseGround(c, Obj(T))$, where c is a clause; C is the set of clauses in T ; Obj returns the set of objects present in the fully grounded clauses of a theory; $ClauseGround$ returns the set of clauses that can be produced by performing all combinations of mapping between the variables of a clause and a set of objects. Because there is no means to create objects through clauses in our language, $Obj(T)$ is constant over the entire grounding translation process.

20.3.3 Translation to MaxSAT

We can now define function $CNF(G)$ that translates a fully grounded theory in our language into a MaxSAT instance. MaxSAT instances consist of a Boolean expression, E, in CNF over a set of Boolean variables, V. The disjunctions in E are called MaxSAT clauses. Each MaxSAT clause is associated with a weight that is a positive number or infinity. A *MaxSAT model* consists of truth assignments over the elements of V. A MaxSAT clause is broken iff all of its disjuncts are assigned to false. The cost of a MaxSAT model is a function of the MaxSAT clauses that are broken due to its truth assignments.

Let G be the grounded theory to translate and S be the equivalent MaxSAT instance. We proceed by creating variables in S's V set that correspond to every unique

literal that appears in the theory clauses of G. Each theory clause in G can be treated as a weighted implication. Therefore, theory clauses in G are equivalent to weighted disjunctions of the form: $\neg A_1 \vee \cdots \vee \neg A_m \vee C_1 \vee \cdots \vee C_n[w]$, where w, A_i, and C_i have the same definition as given in the Language section. For every theory clause, C, in G, we create a MaxSAT clause in S that consists of the Boolean variables in V that correspond to the literals in C. The MaxSAT clauses can then be joined together to form S's E expression.

Definition. A grounded clause is *broken* iff all its antecedents are assigned true and all of its consequents are assigned false. A broken clause has a cost equal to its w value.

Definition. A *model of a theory*, T, is a truth assignment over all of the literals that appear in $Ground(T)$.

A partial model is an incomplete truth assignment. The cost of a model is a function of the theory clauses that are broken by its truth assignments. If a model contains assignments that break a hard theory clause, then that model is contradictory.

3.4 Traditional Cost Function

Since we have shown that theories in our language are equivalent to MaxSAT instances, we can use this translation to demonstrate why the cost function used by existing MaxSAT algorithms does not properly take object equalities into account. All existing MaxSAT algorithms use the following function to calculate the cost of a model, M: $TradCost(M) = \sum_{i=1}^{n} Weight(\, Broken(T(M), M)_i)$, where T returns the theory of a model; $Broken$ returns the set of clauses in a theory that are broken in a given model; $Weight$ returns the weight of a broken clause; n is the number of clauses returned by $Broken$.

To understand why this cost function is problematic, consider the following theory, T_G: $\{P(a1)[\infty],\ P(a2)[\infty],\ a1 = a2[\infty],\ \neg Q(b)[3],\ P(a1) \Rightarrow Q(b)\ [2],\ P(a2) \Rightarrow Q(b)\ [2]\}$. Note that the first three clauses force the following assignments in all non-contradictory models of T_G: $P(a1) = \mathbf{true}$, $P(a2) = \mathbf{true}$, and $a1 = a2 = \mathbf{true}$. The only literal free to assign is $Q(b)$. Consider two models of T_G: $M1$ and $M2$. $M1$ is the model where $Q(b) = \mathbf{true}$. $M2$ is the model where $Q(b) = \mathbf{false}$.

With the formulation from above, $TradCost(M1) = 3$, because $\neg Q(b)[3]$ is broken by $M1$'s assignments and $TradCost(M2) = 4$, because $P(a1) \Rightarrow Q(b)$ [2] and $P(a2) \Rightarrow Q(b)$ [2] are broken by $M2$'s assignments. Thus, existing approaches would select $M1$ as the better model. However, $P(a1)$ and $P(a2)$ can be thought of as "evidence" for $Q(b)$ to be true. Intuitively, it seems that $M2$ is the better model, because if two pieces of evidence for something are identical, they should not independently support it.

20.3.4 New Cost Function

The traditional cost function does not work because it counts the weights from each broken clause regardless of whether or not that clause is redundant because of equality relations. Ideally, a cost function would identify and correct for such redundancies. To help define our new cost function, we introduce the ideas of literal and clause equivalence within a model, M, of a theory, T.

Definition. Two literals, $L1$ and $L2$, are *equivalent* in M iff they have the same predicate, the same number of arguments and every i^{th} argument in $L1$ is either identical to or appears with the i^{th} argument of $L2$ in an equality relation in T that has been assigned to true in M. For example, the following literals: $P(a1)$ and $P(a2)$ are equivalent in M if the following assignment occurs in M: $a1 = a2 = \mathbf{true}$.

Definition. Two clauses, $C1$ and $C2$ are *equivalent* iff they have the same weight, they have the same number of antecedents and consequents, and every i^{th} literal in $C1$ is equivalent to the i^{th} literal in $C2$. [69] For example, the clauses $P(a1)[\infty]$ and $P(a2)[\infty]$ are equivalent in M if the following assignment occurs in M: $a1 = a2 = \mathbf{true}$. Additionally, we define an *equivalence class*, E, of clauses in M as a set of clauses in M's theory, T, that are all equivalent to each other, but to no other clauses in T outside of E. E has a weight identical to the weight of its member clauses. We say that E is broken in M if its member clauses are broken in M.

Definition. We define our new cost function as: $Cost(M) = \sum_{i=1}^{n} Weight(\, Broken(Equiv(T(M), M))_i)$, where T returns the theory of a

[69] A set of clauses can avoid being classified as equivalent by adding a unique spurious literal to each clause.

model; $Equiv$ returns the set of equivalence classes in a theory given a model of that theory; $Broken$ returns to the set of broken equivalence classes in a set of equivalence classes; $Weight$ returns the weight of an equivalence class.

We can see how this function allows for correct inference with T_G by using it to compute the costs of $M1$ and $M2$ from section 21.3.4. Note that $P(a1) \Rightarrow Q(b)$ [2] and $P(a2) \Rightarrow Q(b)$ [2] form an equivalence class for $M1$ and $M2$, because $a1 = a2$ is assigned to true in both models. Thus, $Cost(M1) = 3$ and $Cost(M2) = 2$, which is consistent with our expectations.

20.4 Finding Models

Our goal is to find a model of a theory with the lowest possible cost. We will now introduce a new systematic "lazy instantiation" DPLL-based algorithm for finding such models. It operates on the MaxSAT translation of theories and uses our new cost function. We show that the branch-and-bound heuristic cannot be used with this algorithm. A replacement heuristic is given as an alternative.

20.4.1 Lazy Instantiation

Lazy instantiation can potentially save enormous amounts memory and time during the model finding process (Singla & Domingos, 2006). This technique is based on the observation that often a large number of the variables that occur in a MaxSAT problem are "irrelevant." For example, the MaxSAT translation of the theory, T_{RR}, contains many irrelevant variables. Consider that $Ground(T_{RR})$ produces the following grounding of $EchoOf(?snd1, ?snd2) \Rightarrow ?snd1 = ?snd2[\infty]$: $EchoOf(player1, shot) \Rightarrow player1 = shot[\infty]$. This grounding would ultimately result in the addition of a variable in the MaxSAT translation of T_{RR} that corresponds to the literal $EchoOf(player1,\ shotSound)$. Intuitively, this literal is not relevant and should not be considered.

Formally, a MaxSAT variable is relevant[70] iff it corresponds to a literal that: a.) appears in a fact or b.) appears in a grounded clause that could be broken due to new

[70] The concept of relevant variables is similar to that of "active atoms," which is discussed in (Singla & Domingos, Memory-efficient inference in relational domains, 2006).

assignments in the current model. Such clauses are called relevant clauses. The facts in a theory provide the initial set of relevant variables and clauses. As assignments are made to relevant variables, new variables and clauses become relevant.

20.4.2 Main Model Finding Algorithm

We now present a MaxSAT solving algorithm that uses our cost function. MaxSAT algorithms find lowest cost models by examining the costs that result from different truth assignments. Our algorithm is based on DPLL (Davis, Logemann, & Loveland, 1962), which is a well-known systematic algorithm. Systematic algorithms work by producing successively larger models in a regular fashion. DPLL does this by picking an unassigned variable from the current partial model being examined and creates two "branches" in search space: one where that variable is assigned to true and one where that variable is assigned to false.

The $DPLL-EQUAL$ function, shown in Algorithm 1 (depicted in Figure 56), takes as input a theory, t, and outputs either a model for that theory with the lowest possible cost or $Fail$. Inside $DPLL-EQUAL$, three new variables are declared: m, which is set to an empty model (i.e. one with no assignments); q, which is set to an empty queue of literals; f, which is set to the set of fact clauses that appear in t. For every fact, f_i, an assignment is created in m for f_i's literal according to f_i's valence (i.e. whether or not its literal is negated) and f_i is put in q. $DPLL-EQUAL$ returns the value generated by executing *DPLL-RECUR* on t, m and q.

Algorithm 1: $DPLL-EQUAL(t)$

$m \leftarrow$ EMPTY_MODEL
$q \leftarrow$ EMPTY_LITERAL_QUEUE
$f \leftarrow$ **call** $Facts(t)$
for i **from** 0 **to call** $Size(f)$ **do**
    $m \leftarrow$ **call** $Assign(m,$ **call** $Lit(f_i),$ **call** $Val(f_i))$
    $q \leftarrow$ **call** $Queue($**call** $Lit(f_i))$
endfor
return call $DPLL-RECUR(t, m, q)$

Figure 54. The DPLL-EQUAL algorithm.

$DPLL - RECUR$ (depicted in Figure 57) searches for and returns the best model of t or $Fail$, which indicates that no consistent model exists. We define $Cost(Fail) = \infty$. The arguments of $DPLL - RECUR$ are: t, a theory; m, a model; q, a queue of literals.

First, the $Elaboration$ procedure is called. $Elaboration$ infers new assignments given the existing assignments in m and the clauses in t. DPLL usually employs unit-propagation (Davis et al., 1962) for elaboration, but more sophisticated processes are possible. Additionally, elaboration is the point at which newly relevant literals and clauses can be added to t, m, and q.

After $Elaboration$ is completed, the termination conditions are checked. A search stage can be terminated if m is found to either be contradictory (by calling $Contradictory$) or complete (by calling $Complete$). A model is complete if it contains an assignment for all of the relevant literals in t.

If a search stage is not terminated, then the next literal to branch on, n, is selected by calling $Next$, which returns the element at the front of q. New models are made by calling $Assign$ to add an assignment of n to true or to false to m. $DPLL - RECUR$ is then called with each of those models, eventually producing $b1$, the model with the lowest cost in which n is true and $b2$, the model with the lowest cost in which n is false. The costs of $b1$ and $b2$ are compared using our $Cost$ function; whichever of $b1$ and $b2$ has the lowest cost is returned.

Algorithm 2: $DPLL - RECUR(t, m, q)$

---

call $Elaboration(t, m, q)$
if call $Contradictory(m)$**then**
    return $Fail$
else if call $Complete(m)$ **then**
    return $assign$
endif
$n \leftarrow$ **call** $Next(q)$
$newM \leftarrow$ **call** $Assign(m, n,$ **true**$)$
$b1 \leftarrow$ **call** $DPLL - RECUR(c, newM, q)$
$newM \leftarrow$ **call** $Assign(m, n,$ **false**$)$
$b2 \leftarrow$ **call** $DPLL - RECUR(c, newM, q)$
if call $Cost(b1) <$ **call** $Cost(b2)$ **then**
    return $b1$
else
    return $b2$

endif

Figure 55. The DPLL-RECUR algorithm.

20.4.3 Branch-and-Bound Heuristic

Other weighted DPLL algorithms use a heuristic called "branch-and-bound," which prevents the exploration of search branches that cannot have a lower cost than the best complete model discovered so far. To implement branch-and-bound, all that is required is to keep a record of the best complete model found so far, b, and to add a new termination condition that returns from a search stage if the current model has a higher cost than b. This practice is able to eliminate the unnecessary exploration of many branches, thus greatly reducing the search time.

In order to be complete, branch-and-bound requires that the costs of models increase monotonically as new assignments are added to those models. The traditional cost function, which existing implementations of DPLL use, guarantees that model costs are monotonic. However, if it were possible for new assignments to reduce the cost of the current partial model, then any model whose cost exceeds the lowest known model cost at some point could potentially have its cost reduced in subsequent search stages.

It is not possible to implement branch-and-bound for our algorithm because our cost function does not lead to monotonic model costs. For example, assume clauses $c1$ and $c2$ each have weight 10 and are broken in model M. They contribute 20 to the cost of M. If an equality relation that makes them equivalent is later assumed to be true, then the total weight contributed by $c1$ and $c2$ is reduced to 10.

20.4.4 Equality-First Heuristic

Although equality invalidates the branch-and-bound heuristic, it can enable new heuristics. To illustrate, we created a new heuristic for our algorithm that increases its efficiency. One trend we observed is that gaining information about the equality of clauses seems to reduce the number of branching decisions during search. This occurs because the best values of literals in potentially equal clauses are strongly related.

Thus, our heuristic entails branching on literals with equality predicates before any other types of literals are examined. We call this heuristic "equality-first."

Equality-first can be implemented by changing the $Next$ function in $DPLL-RECUR$ to search the queue for equality literals. If an equality literal is found, then it is returned. Otherwise, the first element of the queue is returned. Because it only changes the order in which literals are branched on, equality-first is a type of variable-ordering heuristic.

20.5 Evaluations

We conducted experiments with an implementation of our algorithm to investigate the equality-first heuristic. These experiments were based on randomly generated theories in our language. Theories were generated based on a set of variable parameters, including: the number of predicates and objects that could appear in literals, the number of initial facts, and the number of constraints. For every generated theory, we compared the time required to run our algorithm with and without the equality-first heuristic enabled.

A number of issues had to be considered during this process. For instance, because the ordering of variables has a very large effect on the performance of DPLL (Davis et al., 1962), we randomized the order in which literals were obtained from the queue (except for equality literals when the heuristic was used). Thus, we could control for this variation by running our algorithm on each theory multiple times and using the average of those runs as the estimated run time for that theory. Additionally, we decided to exclude theories if: only one model existed for the theory, an equality literal was never selected from the queue, or only equality literals were selected from the queue. An execution time limit of 10 minutes was also enforced for each theory. If our algorithm spent more than 10 minutes to run a given theory in either mode, then that theory was excluded.

Our experiment was based on 100 theories randomly generated according to a set of parameter values. Of those theories, 85 met our criterion to be included in the final analysis. Our algorithm ran each of the 100 theories 10 times. We recorded the average run times for those trials. Since our algorithm solved a large number of the theories in

less than one second, we chose to report results only from the 15 theories that took the longest to run. Table 4 contains these results sorted so that theories with fewer relevant clauses appear higher in the table. The first column contains the names of theories, t. The second and third columns contain the number of relevant clauses, c, and literals, l, in each theory. The fourth and fifth columns contain the average run times in milliseconds without, r_w, and with, r_h, the heuristic. The sixth column, p, contains the percentage by which the run times were decreased by using the heuristic.

For almost every theory, the algorithm was able to find models faster if the heuristic was used. We account for the exceptions as consequences of the large effect that the order of variables has on DPLL's runtime. Nevertheless, it is clear that the use of our heuristic provides an overall advantage. In some cases, the improvement in run time was on the order of minutes. These results suggest that the heuristic not only improves the efficiency of our algorithm across a wide range of theories, but also allows it to find models for some theories that were otherwise intractable.

Table 4. Average run times.

t	c	l	r_w	r_h	p
t-1	72	43	9655	9211	4.8
t-2	88	53	427368	319465	33.8
t-3	92	41	14876	14338	3.7
t-4	98	56	17588	12200	44.1
t-5	109	61	126475	47526	166.1
t-6	110	70	128364	38505	233.3
t-7	111	63	5262	5530	-4.8
t-8	125	68	6312	4280	47.5
t-9	133	44	17091	8549	99.9
t-10	135	68	217455	83071	161.8
t-11	136	78	20265	21893	-7.4
t-12	142	52	10429	8708	19.8
t-13	154	61	38745	19145	102.4
t-14	160	72	81584	54564	49.5
t-15	164	75	36800	49688	-25.9

20.6 Related Work

There is a sizable body of work that deals with extending resolution and satisfiability to include equality. (Morris, 1969)describes how to incorporate equality in a first-order resolution procedure. (Badban, van de Pol, Tveretina, & Zantema, 2004)describes a DPLL-based algorithm that finds models of relational theories with

equality. (Gammer & Amir, 2007)describes a method for translating theories in ground logic with equality into satisfiability problems. (Noureddine, 2005)describes how the properties of equality can lead to improvements in efficiency (i.e. through the relaxation of certain equality constraints). Yet, none of this work investigates how equality affects the cost of models for weighted theories.

A lazy version of DPLL is described in (Poon, Domingos, & Sumner, 2008). However, this work uses the same traditional cost function that we have demonstrated to be insufficient above. This paper does not introduce specific heuristics that can be used to improve the performance of lazy DPLL algorithms.

Additionally, some existing MaxSAT-based approaches have been applied to problems that involve equality. A prime example of this is the use of the Alchemy system to perform entity resolution (Singla & Domingos, 2006). That work claims that object equality for this problem can be handled in the existing Alchemy system by creating an equality predicate for each type of object in the model and then encoding the axioms of equality as clauses over those predicates. This appears to work for entity resolution, but it is important to note that the Alchemy system has the ability to learn the weights on clauses. It may be, in the case of entity resolution, where presumably a small number of entities are equal, that such weight learning counteracts the fact that the weights of models are not calculated correctly. Nevertheless, when we ran alchemy on our Russian roulette example, it did not produce the intuitively correct inference.

20.7 Conclusion

Using MaxSAT-solving to make inferences over relational theories is an effective paradigm for automatic reasoning systems. However, in some cases, ignoring object equalities when calculating the cost of a MaxSAT model can lead to unsound inference. This is problematic, since many important problems deal with object equality. We have addressed these problems by introducing a new language for MaxSAT solving and a new cost function for models in that language. Additionally, we described a lazily instantiated algorithm based on DPLL for finding such models.

Our cost function prevents the use of the branch-and-bound heuristic that is typically incorporated into DPLL. To mitigate this loss of efficiency, we developed a new

heuristic, “equality-first,” that is compatible with our algorithm. We were able to demonstrate through experimentation that our heuristic yields valuable improvements in the time to find models for a wide variety of theories in our language. This demonstrates that the equality structure of theories can be exploited to improve the efficiency of inference.

21. Parsing with Polyscheme and SkyPhrase

At this point, the final remaining task in this dissertation is to compile an outline how a human-level NLP system might be realized. This is a project that is of vital importance both to the fields of Computer Science and Cognitive Science and will certainly have a major impact in many other areas of industry, commerce, research, and personal pursuits. Additionally, NLP is generally regarded as an AI-hard problem, which means, at the very least, that insights into solving NLP problems at the human level will probably lead to insights into novel solution for human-level AI in general. Therefore, there is a large motivation for successfully creating such a system.

To recap, this dissertation has focused on examining various aspects of the human-level NLP problem and the range of computational techniques historically and currently utilized for solving them. It has discussed classical problems NLP such as: part of speech tagging, shallow and deep parsing, and coreference, as well as methods such as: Markov Models, Maximum Entropy Models, and PCFGs that have been deployed to solve these problems. Through this discussion, we have seen arguments for and against the use of purely knowledge-lean and knowledge-rich approaches for AI systems, especially for projects that focus on NLP. The most important commonality between these approaches is that while there have been some notable advances towards solutions to these problems and many refinements to the methods used to solve them, it also the case that the current state-of-the-art is very far from replicating human-level performance.

In order to determine why there is still a large disparity between computer and human capacities to process natural language, the following approach was used. First, the advantages and disadvantages of the most common computational methods for NLP were examined. Second, these advantages and disadvantages were related to the various aspects of the NLP problem from the perspective of studies in linguistics. From this survey, it was concluded that while there are compelling reasons to use either knowledge-rich or knowledge-lean techniques to model NLP tasks, neither tradition alone is sufficient for enabling human-level NLP.

At this point, it is necessary to take stock of what these conclusions mean in terms of projects that seek to create artificial human-level AI systems. Exclusively knowledge-

rich systems are troubled with inflexibility and an inability to efficiently self-adapt (i.e. learn). Exclusively knowledge-lean systems are troubled with lack of depth and inability to efficiently self-correct (i.e. revise beliefs in a structured way) without intervention. Therefore, it is beneficial to combine elements of knowledge-rich and knowledge-lean systems into a single hybrid system, which maintains the depth and belief-revision properties of knowledge-rich systems with the flexibility and powerful learning algorithms associated with knowledge-lean systems.

It is useful to point out here that not all hybrid systems are equally equipped to solve human-level NLP problems. Using Markov Logic as an example, it was demonstrated that limitations in a reasoning system can lead to unsatisfactory results when applied to even basic real-world problems. Fortunately, there is an alternative to a monolithic Markov Logic-like approach—the use of integrated, hybrid, and adaptable algorithms to seamlessly combine knowledge-rich and knowledge-lean structures and representations with a special emphasis on combining specific and general-purpose structures and representations during every step of inference. Through the use of such a system, a platform for reasoning about NLP and other AI-related problems can be created that avoids the state space explosion that accompanies essentially all previous approaches to solving these problems. This enables a system designed with this blueprint in mind to scale up from toy domains to real-world, real-time problem instances.

In order to demonstrate that an adaptive, integrated, and hybrid system is capable of handling the computational demands required to solve human-level NLP and AI problems, this document focused on work done with the Polyscheme Cognitive Architecture. However, the work discussed above was limited only to very specific inference tasks that relied on realized implementations of aspects of Polyscheme's underlying human cognitive substrate theory (specifically work was shown for supporting inferences about time and identity). It is possible to map the entire inference process of NLP to elements in the human cognitive substrate, which can, in turn, be implemented in the Polyscheme Cognitive Architecture. Possible approaches to this problem have been outlined in detail in previous work (Cassimatis, Murugesan, & Bugajska, 2008) (Murugesan, 2009) that is outside of the scope of this dissertation.

A summary of this mapping of language onto Polyscheme Cognitive Architecture elements, called Specialists, is presented briefly in the following text, which accompanies some illustrative diagrams. Consider how this system might parse the sentence: "I saw the man with a telescope." As will be illustrated below, the process of understanding this sentence relies not only on the specific representations and algorithms that are implemented in individual Specialists, but also on the general-purpose representations and algorithms that are realized by having these Specialists interact via Polyscheme's central processing mechanism, which uses the metaphor of attention-sharing to enable the communication between Specialists.

First, each individual word can be considered as both an object (i.e. a word) and an event (i.e. an utterance). It is therefore possible to create objects in Polyscheme that correspond to the words in the sentence and their corresponding utterances. Since events can be reified in Polyscheme, it is unnecessary to distinguish between the actual word objects and the reified utterance event objects. The utterances have a corresponding phoneme (i.e. the word that is uttered) and an utterance time relative to the beginning of the utterance. These utterance times can also be considered as objects that can be labeled starting with t0, which represents the time the word "I" appears at beginning of the utterance, to t7, which represents the full stop (i.e. the period) at the end of the utterance. Solving this aspect of the parsing problem requires temporal reasoning and the ability to posit new objects during the course of inference.

Second, each utterance event has several non-time components: namely the phonetic structure of the utterance (i.e. the particular word that is either written, spoken, or communicated by some other means) and an object that corresponds to the entity to which the utterance event is referring. This is called the referent. Because the referent objects are represented as objects within the inference engine, it is possible to reason about the identity relationship that hold over the referents. In our sentence, none of the referents are identical, but in a sentence like: "John walked to the store because he needed bread," it is possible that the referents of "John" and "he" are the same entity. Note that there might be ambiguity in the identity relationships between referents. Solving this aspect of the parsing problem requires the ability to hypothesize and resolve identity relationships between objects.

Third, each of the utterances have a category assignment that indicates what role the utterance plays in the sentence. Generally speaking, these categories correspond to standard parts of speech: nouns, verbs, adjectives, adverbs, prepositions, determiners, etc. Assigning parts of speech to utterance events facilitates the resolution of identity relationships between referents. This is case because referents will typically only be equivalent to referents whose utterances have the same parts of speech. For instance, the references of utterances that are nouns will usually only be identical to other references of utterances that are nouns. In the sentence "John walked to the store because he needed bread," the referents of "John" and "he" are much more likely to be identical than the referents of "John" and "walked." Note that it is not impossible for utterances with different part of speech types to have identical referents. "Walked" and "it" can have the same referent in the sentence: "He walked to the store; it took a long time." Here "walked" and "it" refer to the specific act of walking introduced in this sentence. Solving this aspect of the parsing problem requires the ability to assign categories to objects and to reason over those categories. Note that categories themselves are also represented in Polyscheme as an object.

Additionally, categorizing utterances helps to reduce the number of possibilities for the structure of the syntactic parse tree that best fits the sentence currently being processed. Consider that many very common English words are almost completely unambiguous about their parts of speech. For instance, "the" is almost always a determiner, which means that it is expected to bind with a phrase that acts like a noun to create yet another, larger noun phrase. This is not significantly different from other state-of-the-art syntactic parsers, which utilize a pre-processing step during which words are tagged with gold-standard part of speech assignments. However, unlike many other state-of-the-art syntactic parsers, Polyscheme does not assume that the initial part of speech assignment is unimpeachable. In Polyscheme, essentially all propositions are uncertain during reasoning and are only resolved once the best global configuration of propositions is determined. Thus, Polyscheme, through a process of belief revision would be able to parse a very unusual sentence, such as "The The is a band," without relying on any dedicated pre-processing step. By decreasing its reliance on human

intervention during reasoning, Polyscheme is able to exhibit much more flexible behavior than other state-of-the-art approaches.

Fourth, sophisticated pattern matching algorithms embedded within individual Specialists enables Polyscheme to infer a limited amount of new knowledge during each step of inference. This process aids in the process of belief revision by helping to manage issues that arise due to cycles of belief, because it allows the state of the system just prior to the execution of each processing step to be saved and recovered precisely. Additionally, this limited inference process prevents state-space explosions that might occur from repeated recursive applications of a patterns. Furthermore, new objects can be introduced during the course of inference, which, as has been explained in the above sections on time and identity inference, is crucial to performing efficient inference in human-level reasoning domains. These patterns also have a cost associated with them that reflects the calculated probability that the pattern will hold given a particular subset of the knowledge base. The most common sort of pattern is a rule-like constraint with antecedents and consequents. In this case, if the propositions on the antecedent side all hold, then the propositions on the consequents side are expected to hold in proportion with the associated cost of the constraint. An example of a constraint as applied to NLP is that a determiner followed by a noun phrase is likely to result in the creation of another noun phrase:

$$IsA(?x, Determiner) \text{^} IsA(?y, NounPhrase) \text{^} Adjacent(?x, ?y)\ [0.95]$$
$$\rightarrow\ IsA(?z, NounPhrase)\ \ Contains(?z, ?x)\ \ Contains(?z, ?y)$$

This constraint has three notable features. First, the propositions in the constraint contain variables (denoted by arguments prefixed by a "?") and constants. Second, this constraint has a relative cost of 0.95, which means that it is very expensive to break. Third, the consequents side contains an unbound variable, ?z, which results in the creation of a new object when this constrain matches. The sections dealing with time and identity reasoning illustrate how important the ability to introduce new objects during inference is to enabling efficient reasoning about large problem instances.

Fifth, all of the operations of the Polyscheme Specialists describe above (time, identity, categories, and pattern matching) are combined during each step of inference by the Polyscheme central processor, which implements an attentional focus-sharing

algorithm. Through the operation of the Specialists, the total system opinion about individual propositions is updated and suggestions are made about the next propositions for the system to focus on. After Polyscheme's inferential limits have been reached, another specialist tracks all of the propositions that remain uncertain and arranges them to be focused on by the entire system in an orderly way. This process creates an emergent implementation of a structured search similar to the DPLL algorithm (Davis, Logemann, & Loveland, 1962). During this process of searching, different truth assignments to uncertain propositions are explored in various logical world configurations. These worlds are explored according to how their cost compares to the current minimal cost worlds that have been located thus far. Once all viable worlds have been explored, the minimal cost worlds contain truth assignments to propositions that constitute the answer to the inference task at hand.

Below is a diagrammatic summary of how Polyscheme would approach the parsing of the sentence "I saw the man with a telescope." In Figure 58, the system has inferred all that it can about this sentence without engaging the searching mechanism. Inferences that were made primarily from the Time Specialist appear in blue, inferences from the Category Specialist appear in green, inferences from Identity Specialist appear in brown, and inferences from the Pattern Matching Specialist appear in gray. The elements in yellow represent the parts of the problem that remain uncertain after all of the inference has completed during this phase of problem solving. This uncertainty corresponds to ambiguity in the attachment of the prepositional phrase "with a telescope." One possibility for attachment is to join this phrase with "man," which results in an interpretation of the sentence that can be paraphrased as "I saw the man in possession of a telescope." The state of the inference system if this attachment decision is made is depicted in Figure 59. Another possibility for attachment is to attach this phrase with "saw," which results in an interpretation of the sentence that can be paraphrased as "Using a telescope, I saw the man." The state of the inference system if this attachment decision is made is depicted in Figure 60. Finally, Figure 48 depicts how Figures 58, 59, and 60 can be thought of as representing the states of logical search worlds organized into a search tree. Figure 58 corresponds to world w1, Figure 59 corresponds to world w2, and figure 60 corresponds to world w3. Note that w1, w2, and

w3 all have an associated cost. The world with the minimal cost that also has no resolvable uncertainties represents the most likely parse. Thus, w1 cannot be a parse because it contains uncertainties that can be explored by the system. Through this cost assignment and search, Polyscheme is able to incorporate statistical and logical knowledge, thus allowing it to be a true hybrid of knowledge-lean and knowledge-rich systems.

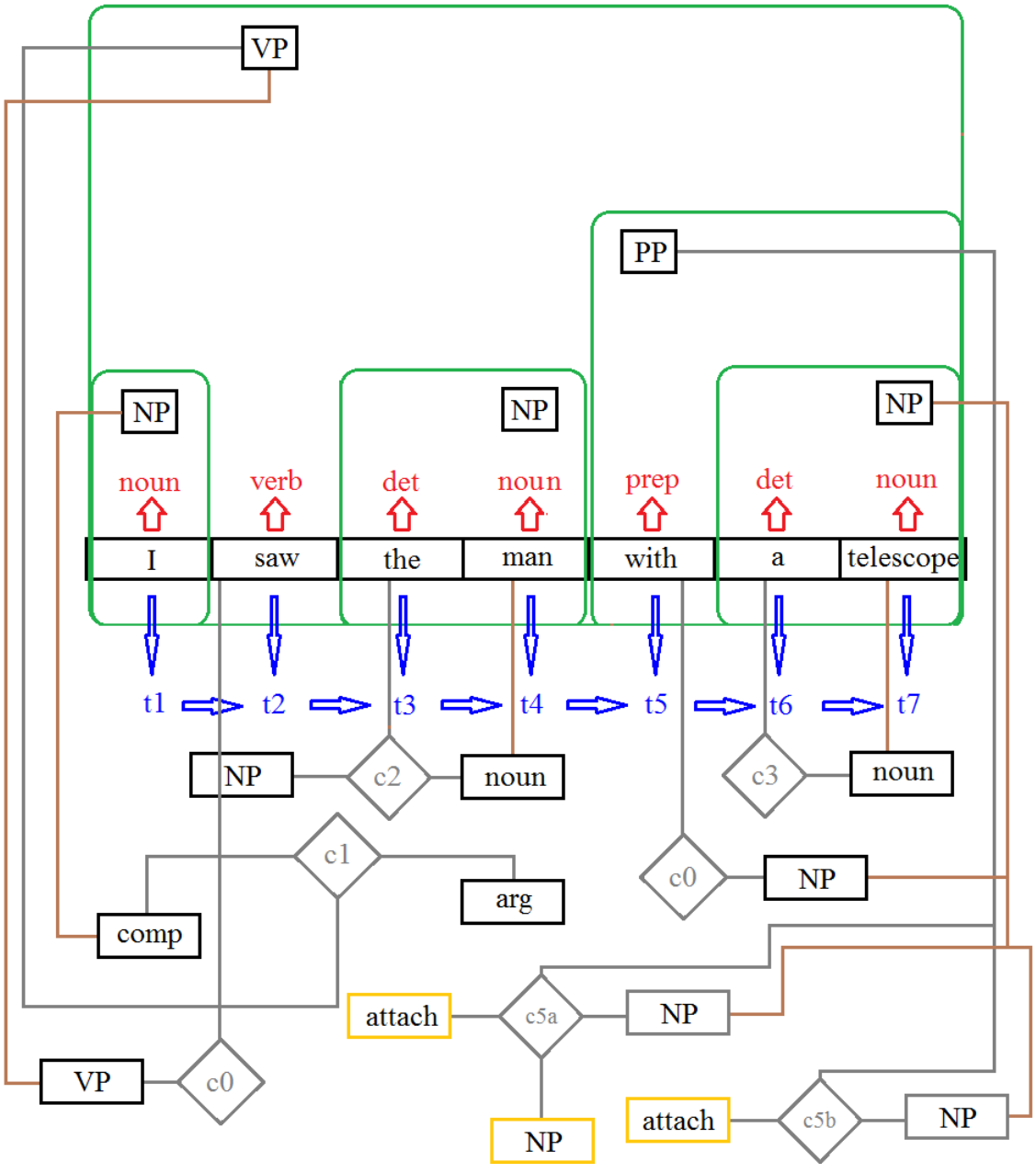

Figure 56. A depiction of Polyscheme representations that have been generated before the prepositional attachment ambiguity has been resolved.

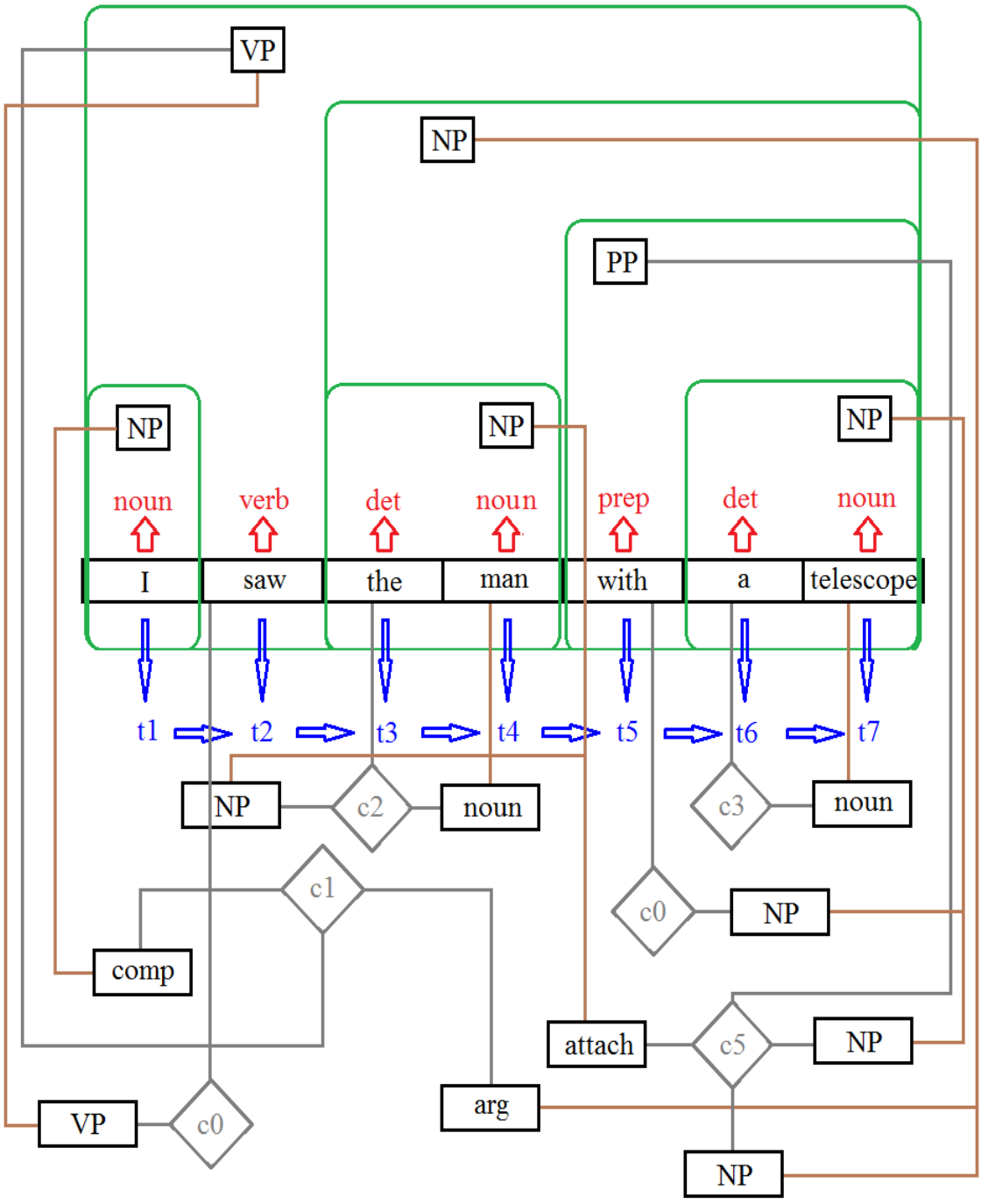

Figure 57. A depiction of Polyscheme representations that have been generated after the prepositional attachment ambiguity has been resolved in favor of attaching it to "man."

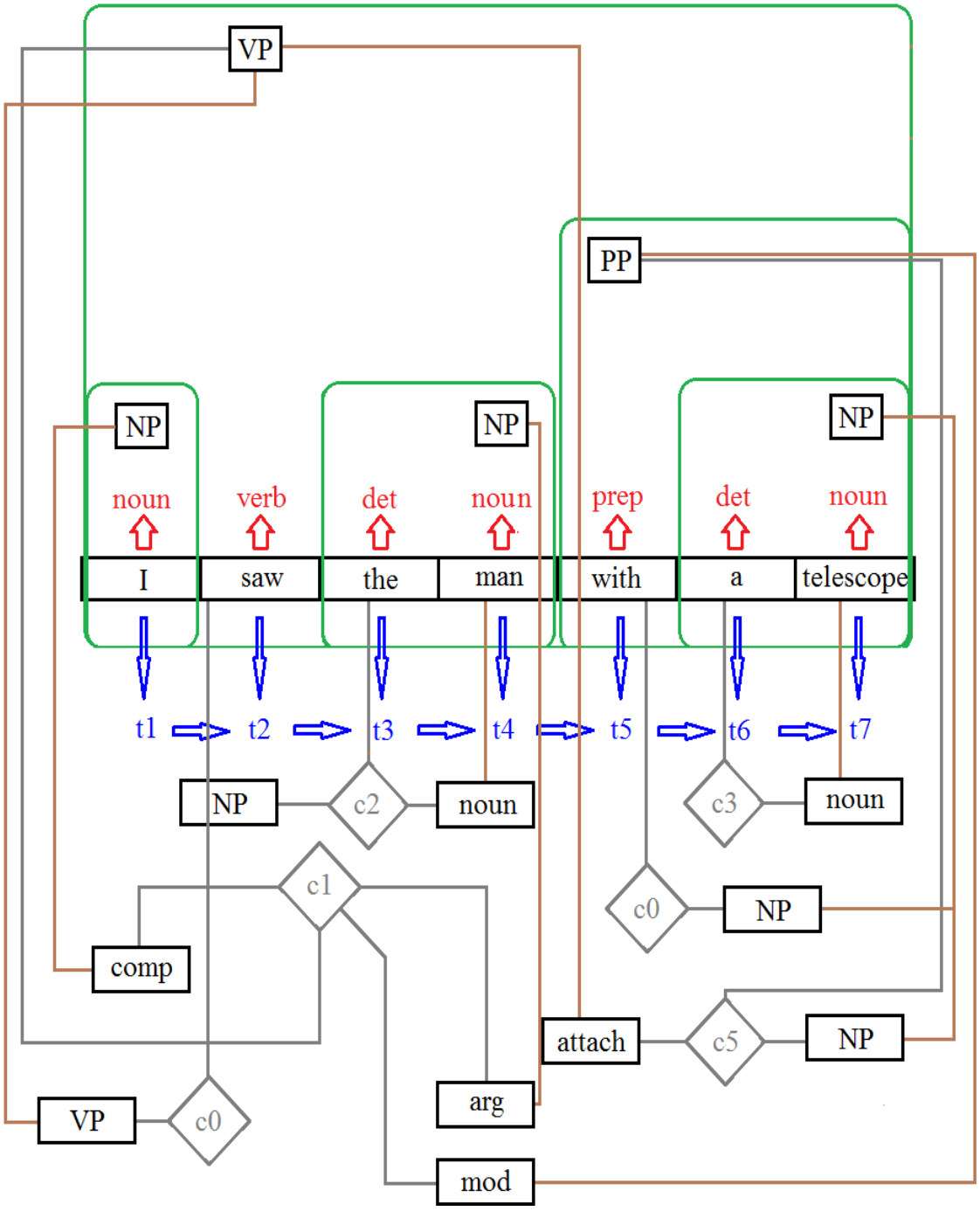

Figure 58. A depiction of Polyscheme representations that have been generated after the prepositional attachment ambiguity has been resolved in favor of attaching it to "saw."

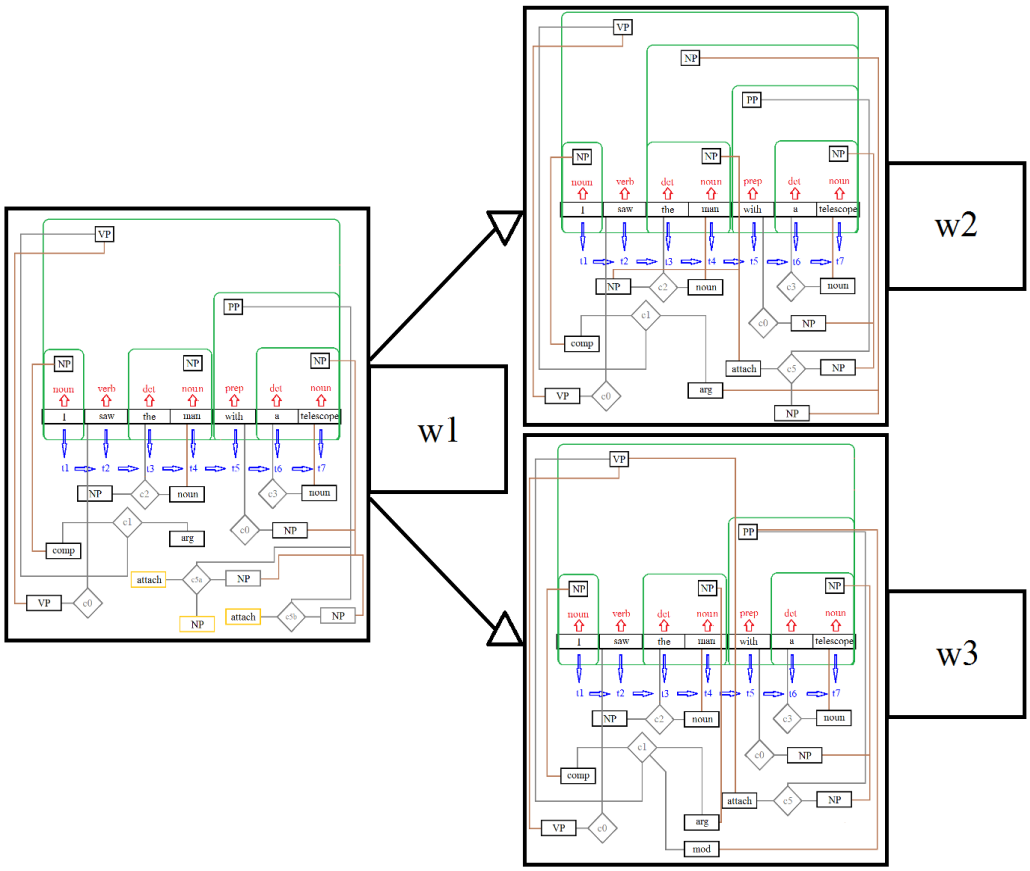

Figure 59. A depiction of how Polyscheme systematically searches for solutions by considering the ramifications of assumptions during the inference process.

One additional point of extreme importance to the Polyscheme parser is that world knowledge is capable of influencing parses in a way that is not possible with other state-of-the-art systems. For instance, if it is known that "I" does not possess a telescope, then the world in which "I" uses a telescope to see "the man" is made much more unlikely, despite what the a priori expectation is for the parse from a purely linguistic perspective. This kind of merging of linguistic and non-linguistic knowledge is possible in Polyscheme because both types of knowledge can be represented using the same structures and operated on using the same algorithms in the same Specialists.

Beyond a Polyscheme-style parser, it is possible to use the fundamental principles of the Polyscheme Cognitive Architecture to create a system that is specifically dedicated to solving human-level NLP problems. The advantage to such a specialized system is that it can be streamlined to perform NLP problem solving much more efficiently than a

general-purpose Polyscheme system. Such a system has been created, which is called the SkyPhrase parser. While the SkyPhrase parser involves algorithms and representations that are inspired by the research presented on efficient handling of Time and Identity inference, it is based on proprietary technology owned by SkyPhrase, Inc. (SkyPhrase, Inc., 2013)that is not discussed in this document.

Although the details of the SkyPhrase parse cannot be disclosed, it is possible to provide a very cursory overview of how SkyPhrase works in comparison to the Polyscheme parser. More details are available in the SkyPhrase Tutorial (Bignoli, 2013). SkyPhrase implements a central, domain-independent parsing system that can be extended with a user-generated model that incorporates domain-specific linguistic knowledge into the total parsing process. This model consists of three components: an ontology, a lexicon, and patterned constructions.

The ontology contains all of the conceptual knowledge involved with understanding the referents of queries in a domain. Entries in the ontology are arranged hierarchically. For instance, the sentence "I saw the man with a telescope" has the following concepts: **Self** ("I"), **Man** ("a man"), **Telescope** ("the telescope"), and a **See** event ("saw"). Higher level concepts join together these low-level concepts: **Self** and **Man** are types of **Person**; **Telescope** is a type of **Tool** and **Tool** is a type of **Object**; **Person** and **Object** are types of **Thing**; **See** is a type of **Action**. Lower level categories inherit from higher-level categories, so **Man**, **Self**, and **Telescope** are also types of **Thing**. Therefore, SkyPhrase's ontology is comprised solely of hierarchical categories that exhibit inheritance and optional disjointness between lateral categories.

The lexicon acts as a map between the actual lexemes that can appear in a domain to the concepts in the ontology associated with the domain. Therefore, there are lexical entries that link **Self** to "I", **Man** to "man", **Telescope** to "telescope", and **See** to "saw." Each of the lexical entries describe textual patterns that may map to particular constructs. In English, these patterns must account for inflected forms of words, so the lexical entry **I** needs to have "I" and "me" as input patterns; the lexical entry **See** needs to have "see," "saw," and "seeing" as input patterns. Items in the lexicon also specify output atoms that indicate what their ontological types are. For example, the lexical entry **I** produces the

atom **IsA Self**. Function words such as determiners (e.g. "the", "a") and prepositions (e.g. "with") are automatically recognized by the SkyPhrase parser.

The patterned constructions control how patterns of lexical entries are assembled into larger syntactic patterns. Patterned constructions also provide an interface between syntax and semantics. For instance, a pattern relevant to the example sentence would be **Thing1_With_Thing2** with the template pattern **N_P_NP**, which corresponds to a prepositional phrase attached to a noun phrase (**N** represents the noun phrase to attach to, **P** represents the preposition, and **NP** is the noun phrase being attached to the noun phrase corresponding to **N**). The input pattern for a patterned construction consists of ontology types, so in this case, the input pattern is: **Thing#1 With Thing#2**. If a pattern construction's input has two or more of the same ontology types, they must be labeled with a '#' followed by a number to ensure that references to those pattern components are unambiguous. Finally, like lexical entries, patterned constructions have an output section that produces atoms that describe the semantics of the construction. **Thing#1 With Thing#2** would output atoms that indicate that **Thing#2** is possessed by **Thing#1**.

A graphical depiction of the components of a SkyPhrase model that would be active in the parsing of "I saw the man with a telescope" is provided in Figure 62. Figure 62 contains all of the relevant ontology entries, lexical entries, and patterned constructions that are necessary for parsing this sentence. Note that pattern constructions must be provided for both possible attachments of the prepositional phrase. **Person_See_Thing1_With_Thing2** handles the case where "with a telescope" attaches to "see;" **Person_See_Thing** and **Thing1_With_Thing2** handle the case where "with a telescope" attaches to "man." SkyPhrase uses statistical information to decide which of the parses is most likely; therefore, like Polyscheme, the SkyPhrase parser is a hybrid of knowledge-lean and knowledge-rich systems.

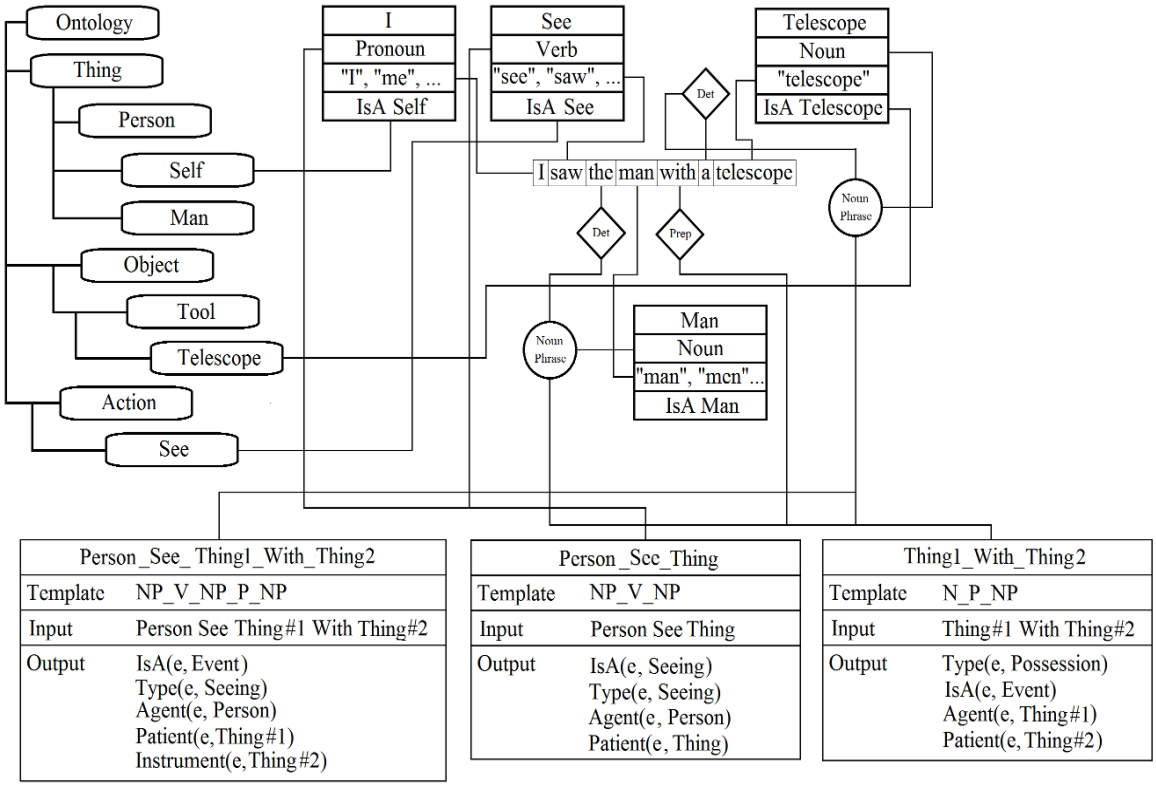

Figure 60. A depiction of the SkyPhrase model elements that are relevant to parsing the sentence "I saw the man with a telescope."

Once a SkyPhrase model is constructed for a specific, reasonably narrow domain, it is coupled with a domain-general SkyPhrase parser, which results in the production of machine-readable, semantic rich output. This output consists of a JSON (Crockford, 2009) object which contains all of the relevant relations that could be extracted from the current utterance being parsed. An example of what this JSON output looks like for the utterance "Chinese Restaurants in Troy" is illustrated below, in an example taken from the SkyPhrase tutorial (Bignoli, SkyPhrase Tutorial, 2013):

```
{
   "query": "Chinese restaurants in Troy, NY",
   "context": "custom",
   "output": {
      "textField_1": {"Text": "troy+ny"},
      "relation_1": {"IsSearch": "true"},
      "condition_1": {"IsA": "restaurants2_ref"},
      "parser": {"SemanticHead": "restaurants2_ref"},
      "restaurants2_ref": {
         "Text": "chinese+restaurants",
```

```
            "Plurality": "plural",
            "IsA": "Restaurant",
            "Location": ["textField_1"]
        }
    }
}
```

The output of a SkyPhrase model is a JSON object, which is delimited by the outermost pair of curly brackets, `{}`. Inside of this object, there are several keys that will show up in the output of every model:

query:

The value is set to be the text string of the query that produced this output. In this case, the query was: **"Chinese restaurants in Troy"**

context:

It is possible to set it up to aid in linking meanings of references between multiple queries. The value of **context** is set based on the previous queries that were supplied to your model, in case if the current query references some part of a previous query. For example, consider if the first query submitted is: **"Restaurants in Troy that serve Chinese food."** Then, if the second query is: "**What time are they open?"** the context would help you determine that "they" refers to the "Restaurants in Troy that serve Chinese food."

output:

The value is a JSON object that has a set of keys that correspond to items that SkyPhrase has been able to extract from the text. For the purposes of this example, the following keys are useful to understanding how to interpret the parse:

parser:

The value contains high-level information about the parse:

```
{"SemanticHead": "restaurants2_ref"}
```

In this case, the output indicates that the semantic head (i.e. **SemanticHead**) of the query is identified with the **restaurants2_ref** element. The semantic head of an utterance is the one that governs its meaning. In this case, the semantic head is the noun that the rest of the utterance is modifying. This noun is "restaurants," which are modified to be "Chinese" and "in Troy, NY."

restaurants2_ref:

The value contains information that the parser has extracted regarding the reference of the word "restaurants" in the input query:

```
"Text": "chinese+restaurants",
"Plurality": "plural",
"IsA": "Restaurant",
"Location": ["textField_1"]
```

Text captures the exact string that is captured as the literal reference. This element is defined automatically by SkyPhrase.

Plurality indicates whether the referent is singular or plural. This element is defined automatically by SkyPhrase.

IsA indicates what ontological types the referent is an instance of. This element is defined by the user in the lexical entry section.

Location indicates the value of the **Location** relation for the reference object. In this case, the value of

Location(restaurants2_ref) is stored in the element called **textField_1**. This element is defined by the user in the patterned construction section.

textField_1:

TextFields contain information that are captured by a **TextField** pattern slot that is defined in a model (there will be a discussion of this below). This **TextField**'s value is the value of **Location(restaurants2_ref)**:

```
{"Text": "troy+ny"}
```

Namely, **restaurants2_ref** is located in **"troy+ny"**

This JSON output object is then read by some kind of processing backend, which handles pragmatic concerns and takes some action based on the nature of the input utterance. For example, if an utterance is a question, an answer may be returned; if an utterance is a command, an action may be performed; if an utterance is a statement, a piece of information might be encoded. Pragmatic concerns might change the kind of response that an utterance receives. For instance, even though the utterance "Can you tell me the time?" is a question with a literal "yes" or "no" answer, pragmatically, this is asking "What time is it?" thus indicating that the user is expecting a time as an answer. Another part of pragmatics handling is to determine the links between different utterances that might have interleaved references. For instance, if a user provides the utterance: "How many people visited my site?" followed by "Last month" and "Last year," the pragmatics handling might decide that the latter two queries are shorthand forms of "How many people visited my site last month?" and "How many people visited my site last year?" respectively.

Another facet of pragmatics handling in SkyPhrase is that successfully responding to an utterance might require vastly different sources of information and interfaces to that information depending on the domain at hand. For instance, in the domain of asking questions about restaurants, it is sufficient for SkyPhrase to implement a simple URL

redirection scheme that automatically takes users to a restaurant information website that contains the information they are interested in receiving. On the other hand, a domain involving sports statistics queries might have to interface with a relational database on the back end. Both of these kinds of solutions have their own engineering-side complications: sometimes it is unclear that a single webpage will contain all of the information that the user desires; sometimes relational databases contain inconsistences that must be addressed to ensure that retrievals produce the expected result.

Like Polyscheme, SkyPhrase needs to be able to combine world knowledge with linguistic knowledge in order to achieve human-level performance. In SkyPhrase, this is accomplished by the creation of domain-specific pragmatics models that operate on the semantic information produced by the output atoms in the corresponding domain-specific parsing model. The pragmatics side interprets what a query actually intends and answers it to the best of its ability. At this point, the SkyPhrase parser executes a precise command to an external data source, which might be a URL call or an access call to a relational database. For instance, when handling "I saw the man with a telescope," SkyPhrase might return a list of men that have telescopes that are within the visual range of the user.

An experiment was conducted to demonstrate how powerful the SkyPhrase parser is in comparison with other state-of-the-art systems. This experiment involved parsing a domain-specific corpus using SkyPhrase and three state-of-the-art commercial systems for NLP: Wolfram|Alpha (Wolfram Research, 2013), Google Search Knowledge Graph (Google, 2012), and Siri (Apple, 2013). The domain selected was information about restaurants that could be found using a site like yelp.com (Yelp, 2013). All four of the platforms used in the experiment were capable of answering queries about the restaurant domain. The experimental corpus, which is included in Appendix 2, consisted of 256 sentences or fragments. These sentences were submitted to the services involved in the study from 17 SEP 2013 to 20 SEP 2013. Since these services are all potentially subject to updates, there may be different results if this experiment is replicated at a later time. Figure 63 and Table 5 contain the results of this experiment. Note that successes were awarded if a reasonable response was returned. For instance, if the query was

"Restaurants near Troy," responses that involved restaurants in Troy, MI or Troy, NY were acceptable, but responses that involved restaurants near Boston, MA were not. Finally, only Wolfram|Alpha reported timeouts during processing. Since it is not known whether Wolfram|Alpha could have produced a correct parse if more processing time was allotted, there are two correctness scores given: one that ignores timeouts and one that assumes timeouts are incorrect.

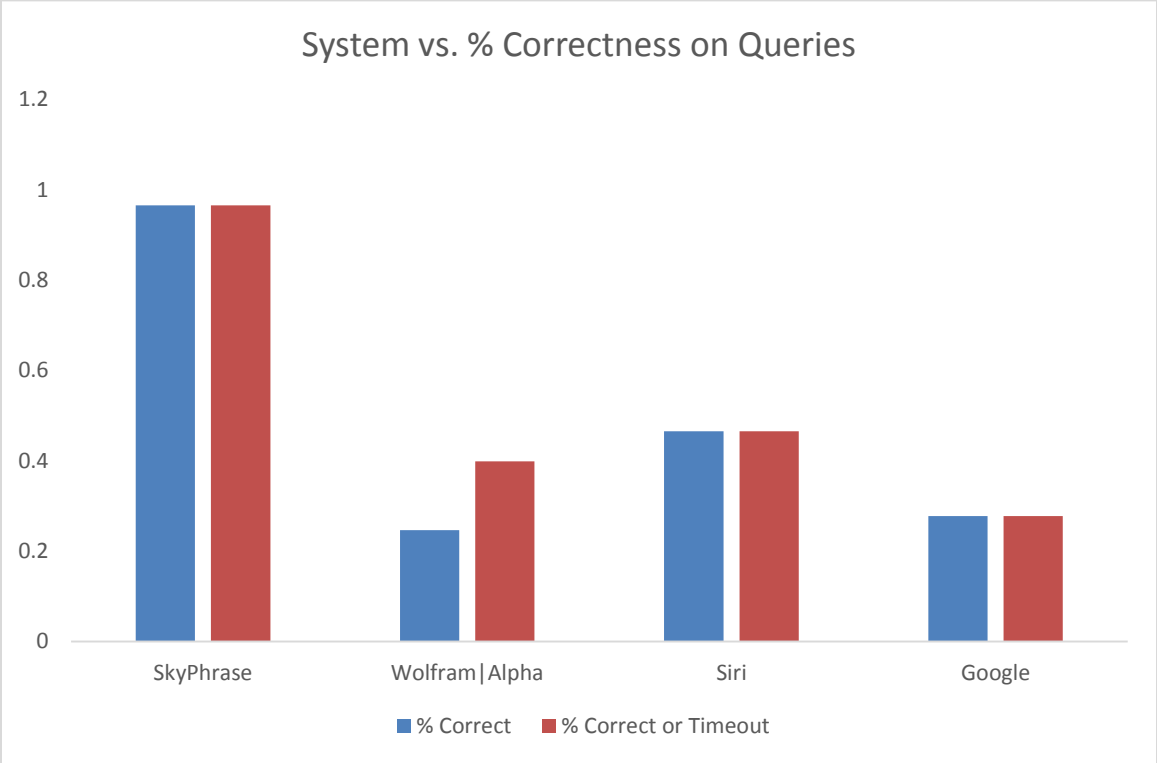

Figure 61. Graph of experimental data for gauging parsing accuracy on a restaurant-domain-related corpus. Note that only Wolfram|Alpha reported processing timeouts.

Table 5. Experimental results of comparing SkyPhrase against Wolfram|Alpha, Siri, and Google on sentences from a restaurant information corpus.

	SkyPhrase	Wolfram|Alpha	Siri	Google
# Correct	247	63	119	71
# incorrect	9	154	137	185
# Time Exceeded	0	39	0	0
Total	256	256	256	256
% Correct	0.96484375	0.24609375	0.46484375	0.27734375
% Correct or Timeout	0.96484375	0.3984375	0.46484375	0.27734375

A second experiment was conducted in order to demonstrate that SkyPhrase was capable of out-performing state-of-the-art systems on multiple domains. This second experiment used a more extensive corpus consisting of 953 utterances from the domain of sports statistics. These queries were presented to Wolfram|Alpha no later than 25 JUL 2013. The sports statistics corpus is included in Appendix 3 of this dissertation. Table 6 below reports the performance of SkyPhrase and Wolfram|Alpha on the utterances in the corpus that are less than 8 words long. Table 7 below reports the performance of SkyPhrase and Wolfram|Alpha on the utterances in the corpus that are 8 or more words long. The shortest utterances in the corpus were 1 word long; the longest utterances were 14 words long. Figure 64 below illustrates how well SkyPhrase and Wolfram|Alpha performed on shorter utterances, as compared to longer utterances. In both length categories, SkyPhrase widely outperforms Wolfram|Alpha. To illustrate how each system fared on individual sentence lengths and to determine if there were any trends in performance given sentence length, Figure 65 was produced. From Figure 65, it is clear that Wolfram|Alpha's performance degrades as the length of query increases. SkyPhrase's performance remains much more consistent.

Table 6. Performance of SkyPhrase vs. Wolfram|Alpha on query utterances less than 8 words in length.

Query Length	SkyPhrase Hit	SkyPhrase Miss	Wolfram|Alpha Hit	Wolfram|Alpha Miss
1	21	0	14	7
2	66	0	34	32
3	106	3	54	55
4	145	4	54	95
5	108	7	35	80
6	97	7	29	75
7	103	10	28	85
Total	646	31	248	429
%	95.4	4.6	36.6	63.4

Table 7. Performance of SkyPhrase vs. Wolfram|Alpha on query utterances greater than 8 words in length.

Query Length	SkyPhrase Hit	SkyPhrase Miss	Wolfram|Aplha Hit	Wolfram|Alpha Miss
8	70	14	16	68
9	68	10	12	66
10	42	10	5	47
11	29	6	2	33
12	11	3	1	13
13	8	0	0	8
14	0	2	0	2
Total	228	45	36	237
%	83.5	16.5	13.2	86.8

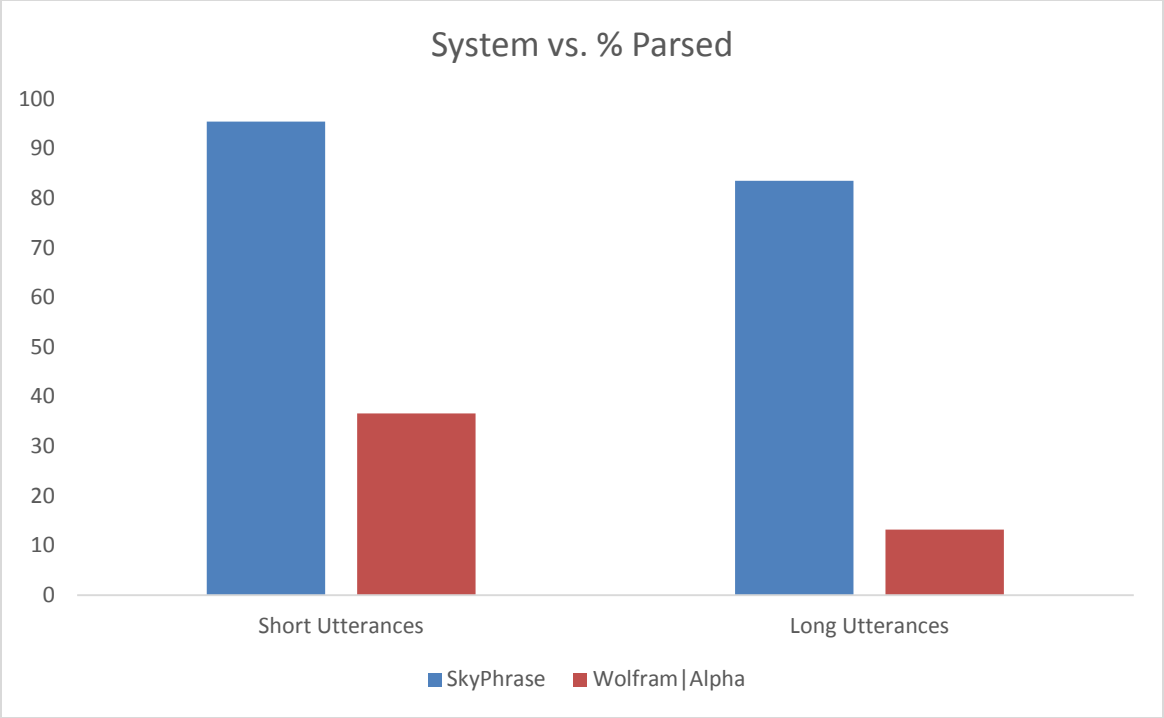

Figure 62. Graph of SkyPhrase vs. Wolfram|Alpha on total sentence accuracy per length category of utterance.

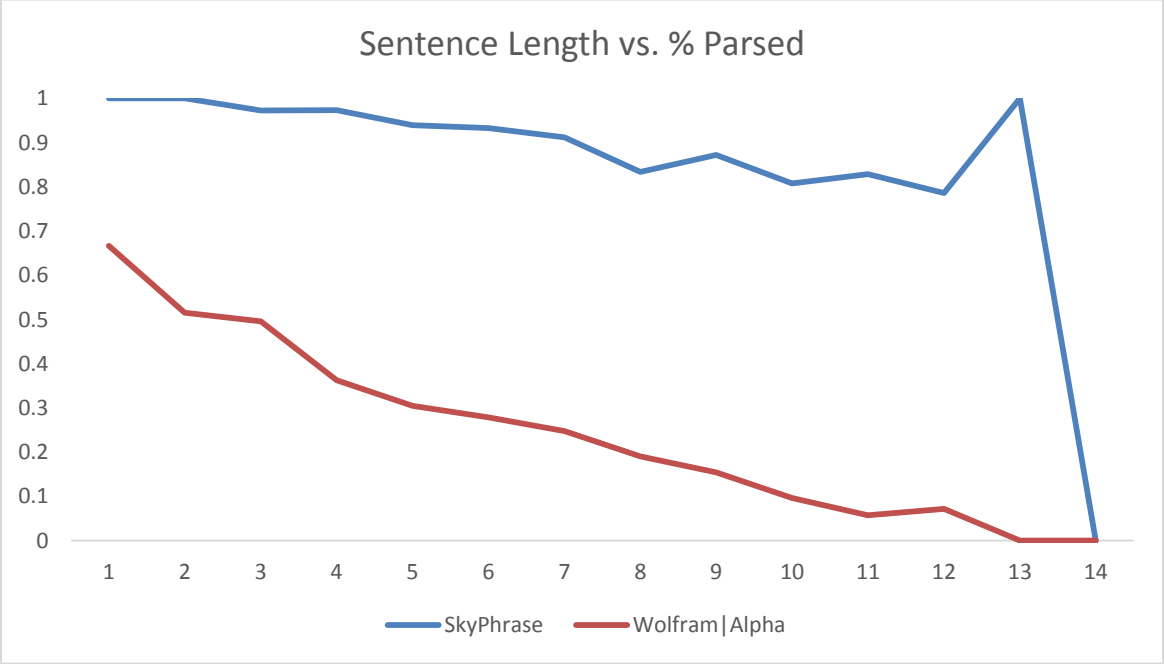

Figure 63. Graph of SkyPhrase vs. Wolfram|Alpha performance on total sentence accuracy per individual utterance length.

From these experiments, it is possible to conclude that the SkyPhrase parser is quantitatively superior to major competing commercial state-of-the-art NLP platforms. This level of accuracy is possible because SkyPhrase, in part inspired by work with the Polyscheme Cognitive Architecture, represents a fruitful combination of knowledge-lean and knowledge-rich inference systems. Moreover, it is capable of leveraging both linguistic and non-linguistic knowledge within the same framework. Additionally, search-based belief revision enables SkyPhrase to be more adaptable and flexible than any other extant system for NLP. Finally, unlike most NLP platforms, SkyPhrase solves the parsing problem at all levels, from part of speech tagging to pragmatics, during one unified inference process. Only minimal, human-free, pre-processing is required for SkyPhrase to parse sentences accurately once an adequate domain-specific language model file has been produced. Even though the corpora used in the experiments above are geared towards answering queries, some of the utterances are commands. SkyPhrase also supports sending alerts when certain condition specified by the user become true. There is no reason in principle why SkyPhrase could not also ingest knowledge in the form of statement utterances. Thus, SkyPhrase is not limited to the NLP subdomain of question answering.

Even though SkyPhrase represents one of the largest leaps towards human-level NLP since the onset of the knowledge-lean NLP era, several major avenue of future research exist. Perhaps the most pressing is that the creation of SkyPhrase domain models and pragmatics handlers must be completely done by humans at this point. In the future, research will be aimed at automating the compilation of certain aspects of these components. Learning some components of SkyPhrase language models are fairly straightforward. For instance, given the large number of ontologies developed for the semantic web and other knowledge-engineering concerns, it should be possible to induct premade ontologies automatically using SkyPhrase. Research is being conducted by several companies, including Google (Singhal, 2012) and Yahoo! (Starr, 2013)to automate the creation of large-scale ontologies even more automatically. Generating lexical entries automatically is more difficult, but plausible, since large dictionaries, thesauri, and concept maps exist in machine readable forms that should enable an

automatic paring of words to concepts, and the automatic association of synonyms (c.f. WordNet (Miller, Beckwith, Fellbaum, Gross, & Miller, 1990)). This leaves the matter of inducing constructions automatically from texts in a given language. Although completing this task will likely require the production of large scale knowledge-rich corpora for learning specific semantic interpretations of constructions, much work has already been done on identifying the most common kinds of constructions in English (Goldberg & Casenhiser, 2006). It is conceivable that similar work could be completed for other languages as well (Goldberg & Casenhiser, 2008). A final concern is that it is probably computationally intractable to implement an exact machine-learning algorithm for the full logical language that a SkyPhrase model describes (Uchida, 2013); however, by focusing on a subset of this full language, it will be possible to at least approach the automatic construction of portions SkyPhrase model through future work.

22. Conclusion

One of the primary goals of both Cognitive Science and Artificial Intelligence is the creation of artifacts that can perform tasks at the human-level of proficiency. As was discussed above, the specific task of natural language processing is extremely important for a number of reasons. The ability to communicate with machines as richly and as fluidly as humans communicate with each other would enable huge advances in technological progress. For instance, consider the implications of being able to perform natural language-driven search queries on scientific documents or having a computer that can distill legalese into basic language. Moreover, understanding how to process language at the human-level would certainly represent a milestone achievement in the broader search for understanding the mechanisms at play in general human-level cognition. There is an overwhelming amount of evidence that the faculties of language are deeply intertwined with most, if not all, other aspects of cognition. Thus, producing models of natural language processing at the human-level will be a major leap forward to producing models of general human-level intelligence.

Despite the huge benefits that would follow from the production of human-level NLP systems, none are presently available, despite over 60 years of dedicated effort towards this goal. Worse yet, there are few major projects currently in development that appear to have any promise for ever achieving real human-level NLP. Despite these facts, there is a major disconnect between the successes reported by both academic and commercial products and the overall qualitative success of the NLP field. This is largely due to an overemphasis on data-driven, knowledge-lean approaches to NLP system architectures.

Several aspects of academic projects have been demonstrated to lead to the continuation of this false impression. Although academic publications have reported human-level performance in some instances, such reports must be carefully reconsidered. First, these reports deal with very narrow and unrealistic portions of the NLP problem, such as Part of Speech tagging, which humans might not even perform during parsing and which humans certainly do not perform in a vacuum, isolated from other components of the NLP task. Second, the metrics used to report these results are skewed towards reporting local success over global success and therefore are highly unfit for gauging the

success of NLP tasks, which invariably rely heavily on global correctness. Third, academic approaches rarely attempt to handle NLP from the bottom-up, starting with completely unstructured text and culminating in semantic and pragmatic level understanding. Those projects that attempt such endeavors, such as Markov Logic, have not been able to scale to real-world, real-time processing applications.

In terms of commercial projects, the outlook is equally dubious. Although commercial may not be as vehemently committed to the knowledge-lean approach as academic projects are, they still do not incorporate truly human-level presentations. Consider that there is a continuous spectrum between knowledge-lean and knowledge-rich, rather than two discrete categories. Commercial products, especially Siri, Watson, and Wolfram|Alpha, might rely more significantly on knowledge than Google projects, but, ultimately, searching over a database of restaurants, trivia facts, or financial data is not a real substitute for the kind of rich, adaptive, and hybridized kinds of knowledge that factor largely in human cognition. There is another major confound when considering NLP from a commercial perspective. At the time that this concluding section is being written, Google has announced that it is creating a Siri-like personal assistant in order to compete for a greater share of the smartphone market. In the NLP domain, commercial innovation looks more like imitation, because companies are only motivated to create products that are either equivalent or only trivially better than their competitors. Since there is such a huge demand for NLP-driven applications and since the performance bar is currently set so low, there is little actual incentive for large commercial concerns to significantly advance the state-of-the-art, so long as their competition remains similarly primitive.

After examining a large number of different and diverse knowledge-lean NLP systems, from Markov Models, Maximum Entropy Models, and Probabilistic Context-Free Grammars (PCFGs) to Support Vector Machines and connectionist models, a number of common themes have become clear. First and foremost of these is that none of these systems actually deliver human-level NLP performance. Any indication otherwise is simply an illusion created by the field-wide usage of misleading metrics that grossly overestimate the capabilities of current systems. Second, these approaches are fundamentally incompatible with the kind of rich, complex world knowledge that is

crucial to all human cognition, especially when it is applied to NLP. For instance, since many of these approaches rely on corpora-based training methods, it is exceedingly difficult to encode this world knowledge in the input data in the first place. Corpora are notoriously expensive to produce by hand, even when they are extremely knowledge-lean (i.e. they only contain Part-of-Speech taggings). Moreover, for some approaches, such as PCFGs, it is difficult to see how complex world knowledge could be leveraged at all, even if it was available in corpora form. Lexicalized PCFGs, which make use of an extremely diluted form of world knowledge, perform scarcely better than standard PCFGs in practice.

A third major issue that is deeply related to the problem of world knowledge is that all of these knowledge-lean approaches are subject to intractably large state-space explosions when they are combined with knowledge-rich representations. As an example, consider the case of Hidden Markov Models. This approach is made vastly more efficient through the use of the Markov Assumption, which limits causal dependencies in the model to be completely local. However, many aspects of semantic and pragmatic knowledge operate on a global scale, at times subsuming the entire discourse. Fourth, the knowledge-lean paradigm fails even on one of the principle motivations for its use—that such methods are purportedly more robust and flexible because they do not rely on hand-coded knowledge structures. This is very deceptive, since all knowledge-lean systems actually rely entirely on hand-coded constraints, which is made clear by the fact that each of the knowledge-lean, probabilistic architectures can be transformed into various kinds of mathematical automata (e.g. Hidden Markov Models are equivalent to finite state transducers). At best, knowledge-lean approaches merely save engineering effort by allowing model designers to specify (by hand) constraint templates that are then populated automatically. However, the use of these templates does not allow knowledge-lean systems to adapt to new conditions or to self-extend by adding new kinds of constraints.

Having dismissed many of the unwarranted claims of the knowledge-lean approaches, it was also necessary to briefly examine the shortcomings of existing knowledge-rich systems. At one point, knowledge-rich systems were the state-of-the-art, before advances in computer memory capacity and processing speed enabled the creation

of large-scale machine-learning-type systems in the early-to-mid-1990s[71]. Although a survey of knowledge-rich approaches reveals that they could match the performance of knowledge-lean approaches on many NLP tasks, they quickly fell out of favor for a number of reasons. Primarily, since knowledge-lean systems are typically much faster to create for a specific problem (mostly because they rely on much simpler hand-coded knowledge and require much less expert linguistic knowledge), they quickly supplanted knowledge-rich approaches. It eventually became common wisdom in the NLP field that knowledge-lean approaches were the architectures of choice.

However, one of the most important findings of this dissertation is that the knowledge-rich approach has many critical properties that are necessary to achieving human-level performance on the full spectrum of NLP tasks. This realization certainly does not preclude the usage of techniques and algorithms that have been developed for knowledge-lean systems. On the contrary, the position developed here advocates for a lessening of the knowledge-rich and knowledge-lean dichotomy in favor of a system that hybridizes the best attribute of both kinds of approaches[72]. Indeed, it is clear that human-level NLP requires complex world knowledge and representations and algorithms that can interact with such structures. These representations and algorithms are relatively expensive to create and maintain and are highly specialized, which is why knowledge-rich systems happened to fall out of favor. History has also demonstrated that the ability to handle the vast amount of statistical knowledge that is available in natural language utterances is just as important. Knowledge-lean approaches evolved to leverage this important source of information with general-purpose, efficient algorithms and representations. Therefore, the only logical conclusion to draw is that the optimal NLP system would involve a hybrid approach that could seamlessly integrate specialized and general-purpose algorithms.

---

[71] Note: one should be particularly wary of algorithms that directly rely on the availability of computing power to increase in power. Although computers are still becoming more powerful, it is well known that the growth factor of this increase is negligible compared to computational complexity of the problems involved in NLP problems.

[72] In a very definite sense, there is actually little underlying difference between machine-learning and expert-system-like approaches. Mainly, it is the way that knowledge is encoded into these systems, rather than the content of that knowledge, that is different. Machine-learning systems are more automatic and template; expert-system are more hand-created and detailed.

Research into Hybrid Cognitive Architectures has culminated in the development of systems that perfectly fit the above requirements set forth for true human-level NLP systems. In particular, the Polyscheme Cognitive Architecture is an extremely promising candidate to serve as the backbone of these systems. Polyscheme was designed to facilitate the full integration of vastly different kinds of algorithms and representations at every step of the inference process. To support this capability, Polyscheme breaks the inference process down into a series of common functions. These common function enable representationally-specific, encapsulated modules called Specialists to communicate with each other during inference and to collectively direct the direction of future inference through a joint focusing of attention. Individual specialists can contain completely arbitrary representations and algorithms, which might be too computationally expensive to implement at the scope of the entire system. Moreover, the joint operation of specialists can lead to the emergence of extremely general algorithms, such as DLPP or WalkSAT, which can capture some of the qualities of knowledge-lean systems. It would even be possible to directly implement a Markov Model-like system as a dedicated Specialist if that turned out to be useful. Hybridization has been shown to greatly increase the computational efficiency of models versus those that exclusively use highly-specific or highly-general types of implementations.

Furthermore, Polyscheme's "canonical" Specialists are constructed exclusively to satisfy the tenets of a Cognitive Substrate Theory, which aims to provide a basis for the entire computational breadth of the human cognitive system with a small number of basic components. These components are all aspects of physical or social reasoning, such as routines for handling reasoning about time, identity, categories, causality, other minds, etc. Many demanding cognitive activities have been decomposed into these relatively few substrate elements. Most importantly for this dissertation, a highly-complex NLP system has been formulated solely through the use of these substrate functions. The major work left to do in Polyscheme was the actual implementation of the individual substrate functions as Specialists. Prior to the work in this dissertation, there were Specialists created for categories, causality, and several other domains. Since reasoning systems for time and identity inference problems were the major remaining components, they were selected for implementation in this study. As was shown above, the models for

time and identity reasoning surpassed the previous state-of-the-art systems on a variety of different problems. These results serve as a proof of concept of the hybridization approach and indicate that this actually promises to be a fruitful avenue of research.

The ultimate test of any proposed technology is whether or not it can actually be implemented in a useful and cost-efficient way. To do this, it was necessary to appeal to the technology developed by SkyPhrase, Inc., which was in part inspired by the work done in Polyscheme. SkyPhrase employs a knowledge-rich model-based approach to creating natural language processors for specific domains. These models are in turn supported by a parser that has encoded many of the possible patterns that are found in English syntax. Together, SkyPhrase models and the SkyPhrase parse engine enable the rapid creation of domain-specific NLP models that exhibit the best qualities of both knowledge-rich and knowledge-lean systems. It has been demonstrated experimentally that SkyPhrase is capable of outperforming state-of-the-art commercial NLP platforms. In the future, one crucial extension to the SkyPhrase system will be to automate parts of the model-building process that currently must be done by humans.

In summation, this dissertation has examined why the current state-of-the-art NLP systems are not only far below the human-level of performance, but also that they are most likely not the correct kinds of formalisms to achieve that goal. After examining academic and commercial systems that utilize either knowledge-rich or knowledge-lean paradigms, a number of common deficiencies were identified. From the results of this survey, a new approach involving the Polyscheme Hybrid Cognitive Architecture was advocated. The hybridization of algorithms for handling inferences over times and identities was demonstrated to provide quantitative and qualitative improvements over non-hybrid systems. By utilizing technology developed by SkyPhrase, Inc. that was inspired by work in Polyscheme, it has been possible to create models of NLP that demonstrably perform qualitatively and quantitatively closer to the human-level of performance than any other existing NLP system.

23. References

Abe, K., & Watanabe, D. (2011). Songbirds possess the spontaneous ability to discriminate syntactic rules. *Nature Neuroscience, 8*(14), 1067-1074.

Abney, S. (1995). Chunks and Dependencies: Bringing Processing Evidence to Bear on Syntax. In *Computational Linguistics and the Foundations of Linguistic Theory* (pp. 145-164). Palo Alto, CA: CSLI Publications.

Abney, S. (1996). Partial Parsing via Finite-State Cascades. *Natural Language Engineering, 2*(4), 337-344.

Abney, S., & Light, M. (1999). Hiding a Semantic Hierarchy in a Markov Model. *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing* (pp. 1-8). Baltimore, MD: ACL.

Ackermann, H., & and Riecker, A. (2004). The contribution of the insula to motor aspects of speech production: a review and a hypothesis. *Brain Language, 89*(2), 320-328.

ACLWiki. (2012, June 25). *Parsing (State of the art).* Retrieved February 20, 2013, from http://aclweb.org/aclwiki/index.php?title=Parsing_(State_of_the_art)

Adler, R. J., & Taylor, J. E. (2007). *Random Fields and Geometry.* New York, NY: Springer.

Ahlswede, T. E., & Lotand, D. (1993). Word Sense Disambiguation by Human Subjects: Computational and Psycholinguistic Applications. *Acquisition Of Lexical Knowledge From Text: Proceedings of a Workshop Sponsored by the Special Interest Group on the Lexicon of the Association for Computational Linguistics.* Columbus, OH: ACL.

Ailliot, P., Thompson, C., & Thomson, P. (2009). Space time modeling of precipitation using a hidden Markov Model and censored Gaussian distributions. *Journal of the Royal Statistical Society, 58*(3), 405-426.

Aimonetti. (2012, February 7). *Siri brings nearly 25 percent of Wolfram Alpha traffic*. (CNet) Retrieved March 10, 2013, from http://reviews.cnet.com/8301-19512_7-57372868-233/siri-brings-nearly-25-percent-of-wolfram-alpha-traffic/

Aitchinson, J. (2001). *Language Change: Progress or Decay?* Cambridge, UK: Cambridge University Press.

Alchemy Research Group. (n.d.). *Alchemy - Open Source AI.* Retrieved February 13, 2013, from http://alchemy.cs.washington.edu/

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM, 26*(11), 832-843.

Altmann, G. (1998). Ambiguity in sentence processing. *Trends in Cogntive Science, 2*(4), 146-152.

Altmann, G., & Mirkovic, J. (2009). Incrementality and Prediction in Human Sentence Processing. *Cognitive Science, 33*(4), 583-609.

Alumae, T., & Kurimo, M. (2010). Domain Adaptation of Maximum Entropy Language Models. *Proceedings of the Association for Computational Linguistics 2010 Conference Short Papers.* Uppsala, Sweden: ACL.

Anderson, C. R., Domingos, P., & Weld, D. S. (2002). Relational Markov Models and their Application to Adaptive Web Navigation. *Proceedings of the Eighth Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining on Knowledge Discovery and Data Mining.* Edmonton, Canada: ACM.

Anderson, J. R. (1996). ACT: A Simple Theory of Complex Cognition. *American Psychologist, 51*(4), 355-365.

Anderson, J. R. (2004). An Integrated Theory of the Mind. *Psychological Review, 111*(4), 1036-1060.

Andras, P. (2002). Equivalence of Support Vector Machine and Regularization Neural Networks. *Neural Processing Letters, 15*(2), 97-104.

Apple. (2013, September 29). *iOS 7*. Retrieved September 30, 2013, from Siri: http://www.apple.com/ios/siri/

Archambeau, C., & Opper, M. (2011). Approximate inference for continuous-time Markov processes. In D. A. Barber, T. Cemgil, & S. Chiappa (Eds.), *Bayesian Time Series Models* (pp. 125-140). Cambridge, UK: Cambridge University Press.

Arkes, H. R., & Ayton, P. (1999). The sunk cost and Concorde effects: Are humans less rational than lower animals. *Psychology Bulletin, 125(5)*, 591-600.

Arslan, B., & Patan, O. (2009). *Developing Methods for Part of Speech Tagging in Turkish.* (Unpublished Senior Project). Bogazici University, Computer Engineering. Istanbul, Turkey.

Association for Computational Linguistics. (2012, October 26). *ACL Wiki*. Retrieved December 23, 2012, from http://aclweb.org/aclwiki/index.php?title=Main_Page

Association of Computational Linguistics. (2013, January 13). *State of the Art*. (ACLWiki) Retrieved February 5, 2013, from http://aclweb.org/aclwiki/index.php?title=State_of_the_art

Aston, G., & Burnard, L. (1998). *The BNC handbook: exploring the British National Corpus with SARA.* Edinburgh, Scotland, UK: Edinburgh University Press.

Augusto, J. C., & Nugent, C. D. (2004). The Use of Temporal Reasoning and Management of Complex Events in Smart Homes. *Proceedings of the Sixteenth European Conference on AI. 16*, pp. 778-782. Amsterdam, Netherlands: IOS Press.

Austin, J. L. (1962). *How to do Things with Words: The William James Lectures delivered at Harvard University in 1955.* (J. O. Urmson, & M. Sbisà, Eds.) Oxford, UK: Clarendon Press.

Autebert, J. M., Berstel, J., & Boasson, L. (1997). Context-Free Languages and Pushdown Automata. In G. Rozenberg, & A. Salomaa (Eds.), *Handbook of Formal Languages* (pp. 111-174). Berlin, Germany: Springer Berlin Heidelberg.

Avesani, R., Azzoni, A., Bicego, M., & Orozco-Alzate, M. (2012). Automatic Classification of Volcanic Earthquakes in HMM-Induced Vector Spaces. In L. Alvarez, M. Mejail, L. Gomez, & J. Jacobo (Eds.), *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications* (pp. 640-647). Berlin, Germany: Springer Berlin Heidelberg.

Baader, F., & Snyder, W. (2001). Unification Theory. In J. A. Robinson, & A. Voronkov (Eds.), *Handbook of Automated Reasoning (Vol. 1)* (pp. 447-533). Amsterdam, Netherlands: Elsevier Science Publishers.

Badban, B., van de Pol, J., Tveretina, O., & Zantema, H. (2004). *Generalizing DPLL and satisfiability for equalities.* Eindhoven, Netherlands: Technical University of Eindhoven.

Baillargeon, R., Spelke, E. S., & Wasserman, S. (1985). Object permanence in five-month-old infants. *Cognition, 20*(3), 191-208.

Ball, J., Freiman, M., Rodgers, S., & Myers, C. (2010). Toward a Functional Model of Human Language Processing. *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society.* Portland, OR: Cognitive Science Society.

Ball, J., Rodgers, S., & Gluck, K. (2004). Integrating ACT-R and Cyc in a large-scale model of language comprehension for use in intelligent agents. *Papers from the AAAI Workshop.* Menlo Park, CA: AAAI Press.

Bangalore, S., & Joshi, A. K. (1999). Supertagging: An Approach to Almost Parsing. *Computational Linguistics, 25*(2), 237-265.

Barner, D., & Bale, A. (2002). No nouns, no verbs: psycholinguistic arguments in favor of lexical underspecification. *Lingua, 112*(10), 771-791.

Barsalou, L. W. (1993). Flexibility, Structure, and Linguistic Vagary in Concepts: Manifestations of a Compositional System of Perceptual Symbols. In A. F. Collins, S. E. Gathercole, M. A. Conway, & P. E. Morris (Eds.), *Theories of Memory* (pp. 29-101). London, England: Lawrence Erlbaum Associates.

Barzilay, R. (2010, November 7). *6.864: Advanced Natural Language Processing*. Retrieved April 10, 2013 from CSAIL at MIT: http://people.csail.mit.edu/regina/6864/

Batzoglou, S. (2007, January 23). *Hidden Markov Models: Decoding and Evaluation [PDF Document]*. Retrieved May 11, 2013 from http://ai.stanford.edu/~serafim/CS262_2007/notes/lecture5.pdf

Baugh, A. C., & Cable, T. (2002). *The History of the English Language.* Oxford, UK: Psychology Press.

BBC News. (2012, March 26). *Google ordered to change autocomplete function in Japan.* Retrieved December 20, 2013, from http://www.bbc.co.uk/news/technology-17510651

Beavers, A. F. (2010). Typicality Effects and Resilience in Evolving Dynamic Associative Networks. *2010 AAAI Fall Symposium Series.* Arlington, VA: AAAI Press.

Beavers, A. F., & Harrison, C. D. (2012). Information-Theoretic Teleodynamics in Natural and Artificial Systems. In H. Zenil (Ed.), *A Computable Universe: Understanding Computation and Exploring Nature as Computation.* Singapore: World Scientific.

Bellugi, U., Wang, P. P., & Jernigan, T. L. (1994). Williams Syndrome: An Unusual Neuropsychological Profile. In S. Broman, & J. Graffman (Eds.), *Atypical cognitive deficits in developmental disorders: Implications for brain function* (pp. 23-56). Hillsdale, NJ: Lawrence Erlbaum.

Bennett, K. P., & Campbell, C. (2000). Support Vector Machines: Hype or Hallelujah? *ACM SIGKDD Explorations Newsletter, 2*(2), 1-13.

Berg, M. (2012). *Human Abilities to Perceive, Understand, and Manage Multi-Dimensional Information with Visualizations*. (Doctoral Dissertation). Aalto University, Espoo, Finland.

Berger, A. (1997, December). *The improved iterative scaling algorithm: A gentle introduction.* Retrieved May 13, 2013 from Carnegie Mellon University School of Computer Science: http://www.cs.cmu.edu/~aberger/pdf/scaling.pdf

Berger, A. L., Della Pietra, V. J., & Della Pietra, S. A. (1996). A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics, 22*(1), 39-71.

Bergmann, R., & Mougouie, B. (2006). Finding Similar Deductive Consequences - A New Search-Based Framework for Unified Reasoning from Cases and General Knowledge. *Proceedings of the 8th European conference on Advances in Case-Based Reasoning* . Ölüdeniz/Fethiye, Turkey.

Bernatzki, A., Eppler, W., & Gemmeke, H. (1996). Interpretation of Neural Networks for Classification Tasks. *Proceedings of EUFIT '96* (pp. 1420-1424). Aachen, Germany: European Congress on Intelligent Techniques and Soft Computing.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American, 284*(5), 28-37.

Bi, J., Bennett, K. P., Embrechts, M., Breneman, C. M., Song, M., Guyon, I., & Elisseeff, A. (2003). Dimensionality Reduction via Sparse Support Vector Machines. *The Journal of Machine Learning Research, 3*, 1229-1243.

Bickerton, D. (1996). *Language and human behavior (The Jessie and John Danz lectures).* Seattle, WA: University of Washington Press.

Bies, A., Ferguson, M., Katz, K., & MacIntyre, R. (1995). *Bracketing Guideline for the Treebank II Style Penn Treebank Project.* Philadelphia, PA: University of Pennsylvania.

Bignoli, P. G. (2012). *Efficient Reasoning about Time in a Domain-General Automatic Inference System: Insights from Cognitive Science Can Illuminate the Path to Human-Level Artificial Intelligence*. (Masters Thesis). Department of Cognitive Science, Rensselaer Polytechnic Institute, Troy, NY.

Bignoli, P. G. (2013). *SkyPhrase Tutorial.* Cohoes, NY: SkyPhrase, Inc.

Bignoli, P. G., Cassimatis, N. L., & Murugesan, A. (2010). Efficient Constraint-Satisfaction in Domains with Time. *Proceedings of the 3rd Conference on Artificial General Intelligence (AGI-2010).* Lugano, Switzerland: Artificial General Intelligence Conferences.

Bikel, D. M., Miller, S., Schwartz, R., & Weischedel, R. (1997). Nymble: a High-Performance Learning Name-Finder. *Proceedings of the Fifth Conference on Applied Natural Language Processing.* Washington, DC: Association for Computational Linguistics.

Bing. (n.d.). *Bing*. Retrieved March 10, 2013, from http://www.bing.com/

Blache, P. (1998). Proof Nets for Controlling Ambiguity in Natural Language Processing. *Proceedings of the Tenth IEEE International Conference on Tools with Artificial Intelligence.* Taipei, Taiwan: IEEE Computer Society Press.

Black, E. (1992, July 17). *Meeting of interest group on evaluation of broad-coverage grammars of English.* Retrieved September 19, 2013, from Linguist List 3.587: http://linguistlist.org/issues/3/3-587.html

Blunsom, P. (2004). Maximum Entropy Markov Models for Semantic Role Labelling. *Proceedings of ALTA 2004: African Language Teachers Association.* Madison, WI: Indiana University Bloomington.

Board on Behavioral, Cognitive, and Sensory Sciences (BBCSS). (2006). 4 Progress in Science. In *A Strategy for Assessing Science: Behavioral and Social Research on Aging.* Washington, D.C.: The National Academies Press.

Boardman, C. (2004). *Pearl's Belief Propagation Algorithm and Loopy Bayesian Networks.* Boston, MA: Northeastern University.

Boas, H. C., & Sag, I. A. (2012). *Sign-Based Construction Grammar.* Stanford, CA: CSLI Publications: Center for the Study of Language and Information.

Boltzmann, L. (1872). Weitere studien über das wärmegleichgewicht unter gasmolekülen. *Sitzungsberichte Akademie der Wissenschaften, 66*, 275-370.

Bordes, A., Usunier, N., Collobert, R., & Weston, J. (2010). Towards Understanding Situated Natural Language. *Journal of Machince Learning Research Workshop & Conference Proceedings, 9*, 65-72.

Brailsford, S. C., Potts, C. N., & Smith, B. M. (1999). Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research, 119*(3), 557-581.

Brants, T. (2000). TnT: A Statistical Part-of-Speech Tagger. *Proceedings of the 6th Conference on Applied Natural Language Processing.* Seattle, WA: Association for Computational Linguistics.

Brants, T., & Franz, A. (2009). *Web 1T 5-gram, 10 European Languages-Version 1.* Philadelphia, PA: Linguistic Data Consortium.

Breiman, L. (1996). Bagging Predictors. *Machine Learning, 24*(2), 123-140.

Breiman, L. (2001). Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science, 16*(3), 199-231.

Brill, E. (1992). A simple rule-based part of speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing (ANLC '92).* Trento, Italy: Association for Computational Linguistics.

Brill, E. (1995). Some advances in transformation-based part of speech tagging: A case study in part-of-speech tagging. *Computational Linguistics, 21*(4), 543-565.

Brill, E., & Mooney, R. (1997). An overview of empirical natural language processing. *AI Magazine, 18*(4), 13-24.

Bringsjord, S., & Arkoudas, K. (2004). The modal argument for hypercomputing minds. *Theoretical Computer Science, 317*(1), 167-190.

Bringsjord, S., & Arkoudas, K. (2004). The modal argument for hypercomputing minds. *Theoretical Computer Science, 317*(1), 167-190.

Bringsjord, S., & Bringsjord, A. (2012). Kludge: The Haphazard Construction of the Human Mind. *Philosophical Psychology, 25*(2), 301-305.

Bringsjord, S., & Clark, M. (2006). For Problems Sufficiently Hard ... AI Needs CogSci. In *Between a Rock and a Hard Place: Cognitive Science Principles Meet AI-Hard Problems, Papers from the AAAI Spring Symposium Technical Report SS-06-02* (pp. 23-26). Menlo Park, CA: AAAI Press.

Briscoe, T., & Carroll, J. (2006). Evaluating the Accuracy of an Unlexicalized Statistical Parser on the PARC DepBank. *Proceedings of the COLING/ACL on Main Conference Poster Sessions.* Sydney, Australia: Association of Computational Linguistics.

Briscoe, T., Carroll, J., Graham, J., & Copestake, A. (2002). Relational evaluation schemes. *Proceedings of the Beyond PARSEVAL Workshop at the 3rd International Conference on Language Resources and Evaluation.* Las Palmas, Gran Canaria: LREC.

Brooks, P., & Center, A. I. (2003). *SCP: A Simple Chunk Parser.* Athens, GA: Artificial Intelligence Center, University of Georgia.

Brown, G. T. (2009). The reliability of essay scores: The necessity of rubrics and moderation. In *Tertiary Assessment and Higher Education Student Outcomes Policy, Practice, and Research* (pp. 43-50). Wellington, New Zealand: Ako Aotearoa - The National Centre for Tertiary Teaching.

Burnard, L. (2002). Where did we go wrong? A retrospective look at the British National Corpus. *Language and Computers, 42*(1), 51-70.

Calude, C. S., & Paun, G. (2004). Bio-steps beyond Turing. *Biosystems, 77*(1-3), 175-194.

Carbonell, J. G., & Gil, Y. (1990). Learning by experimentation: The operator refinement method. In R. S. Michalski, & Y. Kodratoff (Eds.), *Machine Learning: An artificial intelligence approach Volume III* (pp. 191-213). Burlington, MA: Morgan Kaufmann.

Cardie, C. (1996). Embedded Machine Learning Systems for Natural Language Processing: A General Framework. In S. Wermter, E. Riloff, & G. Scheler (Eds.),

Connectionist, statistical, and symbolic approaches to learning for natural language processing (pp. 315-238). New York, NY: Springer.

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E. R., & Mitchell, T. M. (2010). Toward an Architecture for Never-Ending Language Learning. *Proceedings of the 24th Conference on Artificial Intelligence (AAAI 2010) Volume 2 Issue 4.* Atlanta, GA: AAAI Press.

Carnegie Mellon University. (n.d.). *Parse a sentence*. Retrieved March 10, 2013, from http://www.link.cs.cmu.edu/link/submit-sentence-4.html

Carpenter, B. (1991). The Generative Power of Categorial Grammars and Head-Driven Phrase Structure Grammars with Lexical Rules. *Computuational Linguistics, 17*(3), 301-313.

Carroll, J., Briscoe, T., & Sanfilippo, A. (1998). Parser Evaluation: A Survey and a New Proposal. *Proceedings of the 1st International Conference on Language Resources and Evaluation.* Grenada: LREC.

Carroll, L. (1872). *Jabberwocky.* Retrieved January 7, 2013, from http://www.jabberwocky.com/carroll/jabber/jabberwocky.html

Cassimatis, N. L. (2001). *Polyscheme: A Cognitive Architecture for Integrating Multiple Representation and Inference Schemes*. (Doctoral Dissertation). Massachusetts Institute of Technology, Cambridge, MA.

Cassimatis, N. L. (2004). *Grammatical Processing Using the Mechanism of Physical Inference.* Washington, DC: Naval Research Lab Washington DC Center for Applied Research in Artificial Intelligence.

Cassimatis, N. L. (2006). A Cognitive Substrate for Achieving Human-Level Intelligence. *AI Magazine, 27*(2), 45-56.

Cassimatis, N. L. (2006). Artificial Intelligence and Cognitive Science have the Same Problem. *AAAI Spring Symposium on Between a Rock and a Hard Place: Cognitive Science Principles Meet AI-Hard Problems.* Stanford, CA: AAAI Press.

Cassimatis, N. L. (2007). Adaptive Algorithmic Hybrids for Human-Level Artificial Intelligence. In B. Goertzel, & P. Wang (Eds.), *Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms* (pp. 94-110). Amsterdam, Netherlands: IOS Press.

Cassimatis, N. L. (2012). Human-Level Artificial Intelligence Must Be an Extraordinary Science. *Advances in Cognitive Systems 1* (pp. 37-45). Palo Alto: AAAI Press.

Cassimatis, N. L. (2013, April 13). Private Communication.

Cassimatis, N. L., Murugesan, A., & Bignoli, P. G. (2009). Inference with relational theories over infinite domains. *Proceedings of FLAIRS-22.* Sanibel Island, FL: AAAI Press.

Cassimatis, N. L., Murugesan, A., & Bignoli, P. G. (2009). Reasoning as Simulation. *Cognitive Processing, 10*(4), 343-353.

Cassimatis, N. L., Murugesan, A., & Bugajska, M. D. (2008). A Cognitive Substrate for Natural Language Understanding. *Proceedings of the 2008 conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference . 171*, pp. 99-106. Memphis, TN: IOS Press.

Cassimatis, N., Bignoli, P., Bugajska, M., Dugas, S., Kurup, U., Murugesan, A., & Bello, P. (2010). An Architecture for Adaptive Algorithmic Hybrids. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 40*(3), 903-914.

Catoni, O., & Mainguy, T. (2013). *Toric grammars: a new statistical approach to natural language modeling.* Retrieved June 13, 2013, from Computing Research Repository (CoRR): http://dblp.uni-trier.de/db/journals/corr/corr1302.html#abs-1302-2569

Chappelier, J.-C., & Rajman, M. (1998). A generalized CYK algorithm for parsing stochastic CFG. *First Workshop on Tabulation in Parsing and Deduction.* Paris, France: TAPD.

Charniak, E. (2010). Top-Down Nearly-Context-Sensitive Parsing. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing.* Cambridge, MA: Association for Computational Linguistics.

Charniak, E., & Carroll, G. (1994). Context-Sensitive Statistics for Improved Grammatical Language Models. *Proceedings of the National Conference on Artificial Intelligence.* Seattle, WA: John Wiley & Sons, Ltd.

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. *Proceedings of the 43rd Annual Meeting of the*

Association for Computational Linguistics. Ann Arbor, MI: Association for Computational Linguistics.

Charniak, E., Goldwater, S., & Johnson, M. (1998). Edge-Based Best-First Chart Parsing. *Proceedings of the Sixth Workshop on Very Large Corpora.* Montreal, Canada: Association for Computational Linguistics.

Chiang, D., & Scheffler, T. (2008). Flexible Composition and Delayed Tree-Locality. *The Ninth International Workshop on Tree Adjoining Grammars and Related Formalisms.* Tübingen, Germany: University of Tübingen.

Chieu, H. L., & Ng, H. T. (2002). A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. *Proceedings of the National Conference on Artificial Intelligence.* Menlo Park, CA: AAAI Press.

Chinchor, N. A. (1998). *Proceedings of the Seventh Message Understanding Conference (MUC-7).* Fairfax, Virginia: National Institute of Standards and Technology.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory, 2*(3), 113-124.

Chomsky, N. (1969). Quine's Empirical Assumptions. In D. Davidson, & J. Hintikka (Eds.), *Words and objections. Essays on the work of W. V. Quine* (pp. 53-67). Dordrecht, The Netherlands: D. Reidel.

Chomsky, N. (1970). Remarks on nominalization. In J. Roderick, & P. Rosenbaum (Eds.), *Readings in English Transformational Grammar* (pp. 184-221). Waltham, MA: Ginn & Co.

Chomsky, N. (1986). *Knowledge of Language: Its Nature, Origin, and Use.* New York, NY: Praeger.

Chomsky, N. (2002). *Syntactic Structures.* Berlin, Germany: De Gruyter Mouton.

Chow, Y.-L. (1990). Maximum Mutual Information Estimation of HMM Parameters for Continuous Speech Recognition Using the N-Best Algorithm. *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-90).* Albuquerque, NM: IEEE Press.

Chow, Y.-L., & Schwartz, R. (1989). The N-Best Algorithm: An Efficient Procedure for Finding Top N Sentence Hypotheses. *Proceedings of the Workshop on Speech*

and Natural Language. Cape Cod, MA: Association for Computational Linguistics.

Chowdhury, G. (2010). *Introduction to Modern Information Retrieval* (3rd ed.). London, UK: Facet Publishing.

Church, K. W., & Rau, L. F. (1995). Commercial Applications of Natural Language Processing. *Communications of the ACM, 38*(11), 71-79.

Clark, A. (2003). Combining distributional and morphological information for part of speech induction. *Proceedings of the 10th Conference on the European Chapter of the Association for Computational Linguistics-Volume 1.* Budapest, Hungary: Association of Computational Linguistics.

Clark, S., & Curran, J. R. (2009). Comparing the Accuracy of CCG and Penn Treebank parsers. *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers.* Singapore: Association for Computational Linguistics.

Clement, L., Gerdes, K., & Marlet, R. (2011). A Grammar Correction Algorithm: Deep Parsing and Minimal Corrections for a Grammar Checker. *Formal Grammar, 14th International Conference, Revised Selected Papers.* Bordeaux, France: Springer.

Cognition. (2008, May 15). *Why Powerset Misses the Point: Relevancy Matters.* Retrieved March 10, 2013, from http://www.cognition.com/pdfs/Why_Powerset_Misses_the_Point.pdf

Cognition. (n.d.). *Cognition: Giving Technologies New Meaning*. Retrieved March 10, 2013, from http://www.cognitionsearch.com/

Collins, M. (1999). *Head-driven Statistical Models for Natural Language Parsing*. (Doctoral Dissertation). University of Pennsylvania, Philadelphia, PA.

Collins, M., & Brooks, J. (1995). Prepositional Phrase Attachment through a Backed-off Model. *Proceedings of the 3rd Workshop on Very Large Corpora.* Cambridge, MA: MIT Press.

Collins, M., & Roark, B. (2004). Incremental Parsing with the Perceptron Algorithm. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics.* Barcelona, Spain: Association for Computational Linguistics.

Collobert, R., & Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *Proceedings of the 25th International Conference on Machine Learning.* Helsinki, Finland: ACM.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning, 20*(3), 273-297.

Cowie, J., & Lehnert, W. (1996). Information Extraction. *Communications of the ACM, 39*(1), 80-91.

Crammer, K., & Singer, Y. (2002). On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *The Journal of Machine Learning Research, 2*, 265-292.

Crockford, D. (2009, May 28). *Introducing JSON.* Retrieved August 15, 2013, from json.org: http://json.org/

Cromer, R. F. (1991). *Language and thought in normal and handicapped children.* Oxford, UK: Basil Blackwell.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems, 2*(4), 303-314.

Dagan, I., Glickman, O., & Magnini, B. (2006). The PASCAL Recognizing Textual Entailment Challenge. In J. Quinonero-Candela, I. Dagan, B. Magnini, & F. d'Alché-Buc (Eds.), *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment* (pp. 177-190). Berlin, Germany: Springer.

Dahlgren, K. (1988). *Naive Semantics for Natural Language Understanding.* Boston, MA: Kluwer Academic Publishers.

Dalrymple, M. (2011). A very long-distance anaphor? *Proceedings of the LFG11 Conference.* Hong Kong: CSLI Publications.

Daly, R., Shen, Q., & Aitken, S. (2001). Review: Learning Bayesian networks: approaches and issues. *The Knowledge Engineering Review, 26*(2), 99-157.

Darlington, K. (1999). *The Essence of Expert Systems.* London, UK: Prentice-Hall International.

Darroch, J. N., & Ratcliff, D. (1972). Generalized Iterative Scaling for Log-Linear Models. *The Annals of Mathematical Statistics, 43*(5), 1470-1480.

Das, D., & Smith, N. A. (2009). Paraphrase identification as probabilistic quasi-synchronous recognition. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1.* Singapore: ACL.

Dasgupta, S., Papadimitriou, C. H., & Vazirani, U. V. (2006). NP-complete problems. In *Algorithms* (pp. 247-282). New York, NY: McGraw-Hill Higher Education.

Davies, M. (2011). *Google Books (American English) Corpus (155 billion words, 1810-2009).* Retrieved August 1, 2013, from http://googlebooks.byu.edu/

Davis, E. (1990). *Representations of Commonsense Knowledge.* San Fransisco, CA: Morgan Kaufmann Publishers, Inc.

Davis, M., Logemann, G., & Loveland, D. (1962). A Machine Program for Theorem Proving. *Communication of the ACM, 5*(7), 394-397.

de Almeida, G. M., & Park, S. W. (2008). Process Monitoring in Chemical Industries – A Hidden Markov Model Approach. *Proceedings of the 18th European Symposium on Computer Aided Process Engineering - ESCAPE 18.* Lyon, France: Elsevier Science.

Deacon, T. W. (2011). *Incomplete Nature: How Mind Emerged from Matter.* New York, NY: W. W. Norton & Company.

Della Pietra, S., Della Pietra, V., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transaction on Pattern Analysis and Machine Intelligence, 19*(4), 380-393.

Deng, L., Dong, Y., & Platt, J. (2012). Scalable Stacking and Learning for Building Deep Architectures. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012).* Kyoto, Japan: IEEE Press.

Denis, P., & Sagot, B. (2009). Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art POS tagging with less human effort. *Proceedings of Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC 23).* Hong Kong: City University of Hong Kong Press.

DeRose, S. J. (1992). Probability and Grammatical Category: Collocation Analyses of English and Greek. In A. W. Mackie, T. K. McAuley, & C. Simmons (Eds.), *For*

Henry Kucera: Studies in Slavic Philogy and Computational Linguistics (pp. 125-152). Ann Arbor, MI: Michigan Slavic Publications.

Dickinson, M. (2006, Spring). *Probabilistic Parsing.* Retrieved Feb 19, 2013, from http://www9.georgetown.edu/faculty/mad87/06/420/slides/parsing-3x3.pdf

Dinesh, N. J., Lee, I., & Webber, B. (2006). Extracting Formal Specifications from Natural Language Regulatory Documents. *Proceedings of the Fifth International Workshop on Inference in Computational Semantics. 17.* Buxton, England: Association for Computational Linguistics.

Divelbiss, A. W., & Wen, J. T. (1997). A path space approach to nonholonomic motion planning in the presence of obstacles. *IEEE Transactions on Robotics and Automation, 13*(3), 443-451.

Dodd, M., Grant, A., & Seruwagi, L. (2011). *Artificial Intelligence Through the Eyes of the Public.* Worcester, MA: Worcester Polytechnic Institute.

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., & Weischedel, R. (2004). The automatic content extraction (ACE) program-tasks, data, and evaluation. *Proceedings of Fourth International Conference on Language Resources and Evaluation (LREC 2004).* Lisbon, Portugal: European Language Resources Association.

Domingos, P., Kok, S., Poon, H., Richardson, M., & Singla, P. (2006). Unifying Logical and Statistical AI. *Proceedings of the 21st National Conference on Artificial Intelligence.* Boston, Massachusetts: AAAI Press.

Doob, J. L. (1953). *Stochastic Processes.* New York, NY: Wiley.

Dowty, D. R. (1982). Tenses, time adverbs, and compositional semantic theory. *Linguistics and Philosophy, 5*(1), 23-55.

DTREG. (n.d.). *SVM - Support Vector Machines.* Retrieved November 27, 2012, from http://www.dtreg.com/svm.htm

Duan, K.-B., & Keerthi, S. S. (2006). Which Is the Best Multiclass SVM Method? An Empirical Study. In M. Haindl, J. Kittler, & F. Roli (Eds.), *Multiple Classifier Systems* (pp. 278-285). Berlin, Germany: Springer Berlin Heidelberg.

Eberbach, E., & Wegner, P. (2003). Beyond Turing Machines. *Bulletin of the European Association for Theoretical Computer Science (EATCS Bulletin), 81*, 279-304.

Eckert, M., & Strube, M. (2000). Dialogue acts, synchronizing units, and anaphora resolution. *Journal of Semantics, 17*(1), 51-89.

Edlund, J. R. (1995). The Rainbow and the Stream: Grammar as System Versus Language in Use. In S. M. Hunter, & R. Wallace (Eds.), *The Place of Grammar in Writing Instruction* (pp. 89-103). Portsmouth, NH: Heinemann.

Edwards, A. (1976). *An Introduction to Linear Regression and Correlation.* San Francisco: WH Freeman.

Een, N., & Sorensson, N. (2005). MiniSat-A SAT solver with conflict-clause minimization. *SAT 2005 Competition.* St Andrews, Scotland: Eighth International Conference on Theory and Applications of Satisfiability Testing.

Elkan, C. (1995). On Solving the Qualification Problem. *Working Notes of the 1995 AAAI Spring Symposium on Extending Theories of Actions: Formal Theory and Practical Applications.* Stanford, CA: AAAI Press.

Elman, J. L. (1993). Learning and development in neural networks: The importance of starting small. *Cognition, 48*(1), 71-99.

Emmorey, K. (2001). *Language, cognition, and the brain: Insights from sign language research.* Hillsdale, NJ: Lawrence Erlbaum.

Engbert, R., Longtin, A., & Kliegl, R. (2002). A dynamical model of saccade generation in reading based on spatially distributed lexical processing. *Vision Research, 42*(5), 621-636.

Everett, D. L. (2005). Cultural Constraints on Grammar and Cognition in Pirahã: Another Look at the Design Features of Human Language. *Current Anthropology, 46*(4), 621-646.

Fadiga, L., Craighero, L., Destro, M. F., Finos, L., Cotillon-Williams, N., Smith, A. T., & Castiello, U. (2006). Language in Shadow. *Social Neuroscience, 1*(2), 77-89.

Fariselli, P., Martelli, P. L., & Casadio, R. (2005). The posterior-Viterbi: a new decoding algorithm for hidden Markov models improves the prediction of the topology of all-beta membrane proteins. *BMC Bioinformatics, 6*(Supplement 4), S12.

Farkas, J. (1993). Neural networks and document classification. *Proceedings of the 1993 Canadian Conference on Electrical and Computer Engineering.* Vancouver, Canada: IEEE Press.

Feldman, R., & Golumbic, M. C. (1990). Optimization algorithms for student scheduling via constraint satisfiability. *Computer Journal, 33*(4), 356-364.

Feng, F., & Croft, B. W. (2001). Probabilistic techniques for phrase extraction. *Information Processing & Management, 37*(2), 199-220.

Ferreira, F., Christianson, K., & Hollingworth, A. (2001). Misinterpretations of Garden-Path Sentences: Implications for Models of Sentence Processing and Reanalysis. *Journal of Psycholinguistics Research, 30*(1), 3-20.

Fillmore, C. J., Kay, P., & O'Connor, M. C. (1988). Regularity and Idiomaticity in Grammatical Constructions: The Case of Let Alone. *Language, 64*, 501-583.

Fine, S., Singer, Y., & Tishby, N. (1998). The hierarchical hidden markov model: analysis and applications. *Machine Learning, 32*(1), 41-62.

Flynn, A. M. (1985). *Redundant Sensors for Mobile Robot Navigation.* Cambridge, MA: MIT Press.

Fodor, J. A. (1983). *The Modularity of Mind: An Essay on Faculty Psychology.* Cambridge, MA: MIT Press.

Forster, P. G. (1982). *The Esperanto Movement.* Berlin, Germany: Walter de Gruyter.

Frakes, W. B. (1992). Stemming Algorithms. In W. B. Frakes, & R. A. Baeza-Yates (Eds.), *Information Retrieval: Data Structures and Algorithms* (pp. 131-160). Upper Saddle River, NJ: Prentice-Hall.

Francesconi, E. (2010). Legal Rules Learning based on a Semantic Model for Legislation. *Proceedings of the LREC 2010 Workshop on the Semantic Processing of Legal Texts (SPLeT-2010).* Valletta, Malta: European Language Resources Association.

Francis, W. N., & Kucera, H. (1964). *A Standard Corpus of Present-Day Edited American English, for use with Digital Computers.* Providence, RI: Department of Linguistics, Brown University.

Franz, A. (1996). *Automatic Ambiguity Resolution in Natural Language Processing: An Empirical Approach.* Secaucus, NJ: Springer-Verlag New York, Inc.

Freund, Y., Schapire, R., & Abe, N. (1999). A Short Introduction to Boosting. *Journal of the Japanese Society for Artificial Intelligence, 14*(5), 771-780.

Fujimaki, R., & Hayashi, K. (2012). Factorized Asymptotic Bayesian Hidden Markov Models. *Proceedings of the 29th International Conference on Machine Learning.* Edinburgh, Scotland, UK: Omnipress.

Fung, P., Ngai, G., Yang, Y., & Chen, B. (2004). A Maximum-Entropy Chinese Parser Augmented by Transformation-Based Learning. *ACM Transactions on Asian Language Information Processing (TALIP), 3*(2), 159-168.

Gamboa, F., & Gzyl, H. (1990). Linear Programming with Maximum Entropy. *Mathematical and Computational Modeling, 13*(7), 49-52.

Gammer, I., & Amir, E. (2007). Solving satisfiability in ground logic with equality by efficient conversion to propositional logic. *Proceedings of the Symposium of Abstraction, Reformulation, and Approximation (SARA-2007)* (pp. 169-183). Whistler, Canada: AAAI Press.

Gao, J., & Johnson, M. (2008). A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing.* Honolulu, HI: Association for Computational Linguistics.

Garnes, S., & Bond, Z. S. (1976). The relationship between acoustic information and semantic expectation. *Phonologica*, 285-293.

Gendron, M., Linquist, K. A., Barsalou, L. W., & Barrett, L. F. (2012). Emotion Words Shape Emotion Percepts. *Emotion, 12*(2), 314-325.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7*(2), 155-170.

Ghahramani, Z., & Jordan, M. I. (1997). Factorial Hidden Markov Models. *Machine Learning, 29*(2), 245-273.

Gibbs, J. W. (1928). *The Collected works of J. Willard Gibbs.* New York, NY: Longmans, Green, and Co.

Gierl, M. J., & Haladyna, T. M. (Eds.). (2013). *Automatic item generation: Theory and Practice.* London, UK: Routledge.

Gigerenzer, G., & Selten, R. (2001). *Bounded Rationality: The Adaptive Toolbox.* Cambridge, MA: MIT Press.

Gigerenzer, G., Todd, P., & Group, T. A. (1999). *Simple Heuristics that Make Us Smart.* London, England: Oxford University Press.

Gildea, D., & Palmer, M. (2002). The Necessity of Parsing for Predicate Argument Recognition. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics.* Philadelphia, PA: Association for Computational Linguistics.

Giles, C. B., & Wren, J. D. (2008). Large-scale directional relationship extraction and resolution. *BMC Bioinformatics, 9*(11), S11.

Global Language Monitor. (2011, July 23). *Number of words in the English language.* Retrieved April 1, 2013, from http://www.languagemonitor.com/new-words/number-of-words-in-the-english-language-1008879

Gold, E. M. (1967). Language identification in the limit. *Information and Control, 10*(5), 447-474.

Goldberg, A. E. (1999). The Emergence of Argument Structure Constructions. In B. MacWhinney (Ed.), *The Emegence of Language* (pp. 197-212). London, UK: Psychology Press.

Goldberg, A. E. (2003). Constructions: a new theoretical approach to language. *Trends in Cognitive Science, 7*(5), 219-224.

Goldberg, A. E., & Casenhiser, D. (2006). 15 English Constructions. In *The Handbook of English Linguistics.* Malden, MA: Blackwell Publications.

Goldberg, A. E., & Casenhiser, D. (2008). Construction learning and second language acquisition. In *Handbook of cognitive linguistics and second language acquisition* (pp. 197-215). Malden, MA: Blackwell Publications.

Goldberg, A. E., Casenhiser, D. M., & Sethuraman, N. (2005). The role of prediction in construction-learning. *Journal of Child Language, 32*(2), 407-426.

Goldberg, Y., Adler, M., & Elhadad, M. (2008). EM can find pretty good HMM POS-taggers (when given a good start). *Proceedings of ACL-08: HLT.* Columbus, OH: Association for Computational Linguistics.

Goldstein, D. G., & Gigerenzer, G. (2002). Models of ecological rationality: The recognition heuristic. *Psychological Review, 109*(1), 75-90.

Golson, J. (2012, September 24). *Siri Delivering Incorrect Weather Forcasts for Common City Names*. Retrieved August 15, 2013, from MacRumors: http://www.macrumors.com/2012/09/24/siri-delivering-wrong-weather-forecasts-for-common-city-names/

Gong, S., & Xiang, T. (2011). *Visual Analysis of Behaviour.* London, UK: Springer.

Gong, Y. (1995). Speech recognition in noisy environments: A survey. *Speech Communications, 16*(3), 261-291.

Goodman, J. (2002). Reduction of Maximum Entropy Models to Hidden Markov Models. *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann Publishers Inc.

Google. (2010, January 19). *Helping computers understand language.* Retrieved January 31, 2013, from http://googleblog.blogspot.com/2010/01/helping-computers-understand-language.html

Google. (2012, May 16). *Introducing the Knowledge Graph: things, not strings*. Retrieved July 20, 2013, from Google Blog: http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

Google. (2013, March 7). *Google - Official Blog*. (Blogspot) Retrieved March 31, 2013, from http://googleblog.blogspot.com

Google. (2013). *Inside Search - Autocomplete*. Retrieved March 10, 2013, from http://support.google.com/websearch/bin/answer.py?hl=en&answer=106230

Grishman, R., & Sundheim, B. (1996). Message understanding conference-6: A brief history. *Proceedings of the 16th International Conference on Computational Linguistics (COLING).* Copenhagen, Denmark: Association for Computational Linguistics.

Grover, D. L., King, M. T., & Kushler, C. A. (1998). United States Patent No. 5818437, Washington, DC: United States Patent and Trademark Office.

Gruber, O., & Goschke, T. (2004). Executive control emerging from dynamic interactions between brain systems mediating language, working memory and attentional processes. *Acta Psychologica, 155*(2-3), 105-121.

Guha, R. V., & Lenat, D. (1990). Cyc: A Midterm Report. *AI Magazine, 11*(3), 32-59.

Hale, J. T. (2009). Heuristic search in a cognitive model of human parsing. *Proceedings of the 11th International Conference on Parsing Technologies.* Paris, France: Association for Computational Linguistics.

Halevy, A. N. (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems, 24*(2), 8-12.

Hawkins, J., & Blakesless, S. (2005). *On Intelligence.* New York, NY: St. Martin's Griffin.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning, 20*(3), 197-243.

Hegner, S. J. (1997). A Family of Decidable Feature Logics which Support HPSG-Style Set and List Constructions. In *Logic Aspects of Computational Linguistics* (pp. 208-227). Berlin, Germany: Springer Berlin Heidelberg.

Heras, F., Larrosa, J., & Oliveras, A. (2007). MiniMaxSAT: a new weight Max-SAT solver. *International Conference on Theory and Application of Satisfiability Testing* (pp. 41-55). Ann Arbor, MI: SAT.

Herken, R. (1995). *The universal Turing machine: a half-century survey.* Vienna, Austria: Springer Verlag Wien.

Herman, L. M., & Uyeyama, R. K. (1999). "The dolphin's grammatical competency: Comments on Kako (1999). *Learning & behavior, 27*(1), 18-23.

Hernandez-Lobato. (2010). *Balancing Flexibility and Robustness in Machine Learning: Semi-parametric Methods and Sparse Linear Models*. (Doctoral Dissertation). Universidad Auto De Madrid, Madrid, Spain.

Hertwig, R., Fanselow, C., & Hoffrage, U. (2003). Hindsight bias: How knowledge and heuristics affect our reconstruction of the past. *Memory, 11*(4-5), 357-377.

Highfill, B. (2011). *Part of Speech Tagging with Discriminatively Re-ranked Hidden Markov Models.* Stanford, CA: Stanford University Department of Computer Science.

Hilbe, J. (2009). *Logistic Regression Models.* Boca Raton, Florida: CRC Press.

Hill, D. R. (1968). A vector clustering technique. In K. Samuelson (Ed.), *Mechanised Information Storage, Retrieval and Dissemination* (pp. 225-234). Amsterdam, Netherlands: North-Holland.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science, 313*(5786), 504-507.

Hirschman. (1997, July 13). *MUC-7 Coreference Task Definition (Version 3.0).* Retrieved March 10, 2013, from http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/co_task.html

Hoffmann, A. G. (1990). On Computational Limitations of Neural Network Architectures. *Proceedings of the 2nd IEEE Symposium on Parallel and Distributed Processing* (pp. 818-825). Dallas, TX: IEEE Press.

Hofstadter, D. R. (1996). *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought.* New York, NY: Basic Books.

Hogenhout, W. R., & Matsumoto, Y. (1998). Robust Parsing Using a Hidden Markov Model. *Proceedings of the International Workshop on Finite State Methods in Natural Language Processing.* Budapest, Hungary: Association for Computational Linguistics.

Hollingshead, K., Fisher, S., & Roark, B. (2005). Comparing and combining finite-state and context-free parsers. *Proceedings of the 2005 Conference on Human Language Technology and Emperical Methods in Natural Language Processing.* Vancouver, Canada: Association for Computational Linguistics.

Holyoak, K., Gentner, D., & Kokinov, B. (2001). The place of analogy in cognition. In D. Gentner, K. J. Holyoak, & B. N. Kokinov (Eds.), *The analogical mind: Perspectives from cognitive science* (pp. 1-19). Nashville, TN: Bradford Books.

Honan, M. (2001, December 5). *Siri Is Apple's Broken Promise*. (Gizmodo) Retrieved March 10, 2013, from http://gizmodo.com/5864293/siri-is-apples-broken-promise

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (1979). *Introduction to Automata Theory, Languages, and Computation.* Reading, MA: Addison-Wesley.

Hornik, K. (1991). Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks, 4*(2), 251-257.

Human, D. P. (2007). Animal cognition: Continuity and discontinuity. *Proceedings of the National Academy of Science of the United States of America, 104*(35), 13861-13867.

Hume, E. V., & Johnson, K. (Eds.). (2001). *The Role of Speech Perception in Phonology.* San Diego, CA: Academic Press.

Hutchens, J. (1996). *How to pass the Turing test by cheating.* Perth, Australia: University of Western Australia School of Electrical, Electronic and Computer Engineering.

IBM. (2004, February). *IBM Research: Natural Language Processing*. (IBM Research) Retrieved March 10, 2013, from http://www.research.ibm.com/compsci/spotlight/nlp/

IBM. (2006, March 16). *IBM to Acquire Language Analysis System, Inc.* Retrieved December 13, 2012, from http://www-03.ibm.com/press/us/en/pressrelease/19394.wss

IBM. (n.d.). *What is Toronto???* Retrieved March 10, 2013, from http://www-03.ibm.com/innovation/us/watson/related-content/toronto.html

Ide, N., & Brew, C. (2000). Requirements, Tools, and Architectures for Annotated Corpora. *Proceedings of Data Architectures and Software Support for Large Corpora.* Paris, France: European Language Resources Association.

Ide, N., & Veronis, J. (1998). Word Sense Disambiguation: The State of the Art. *Computational Linguistics, 24*(1), 2-40.

Iordanova, B. N. (2003). How hierarchies of objects and constraints reduce complexity. *Proceedings of the International Joint Conference in Neural Networks (IJCNN).* Portland, OR: IEEE Press.

Jackendoff, R. (1993). *Patterns in the Mind: Language and Human Nature.* New York, NY: Havester Wheatsheaf.

Jacquemin, C. (1999). Syntagmatic and paradigmatic representations of term variation. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics.* College Park, MD: Association for Computational Linguistics.

Jager, G. (2007). Maximum Entropy Models and Stochastic Optimality Theory. In *Architectures, rule, and preferences: a festschrift for Joan Bresnan* (pp. 467-479). Stanford, CA: Jane Simpson Center for the Study of Language and Information.

Jaynes, E. T. (1957). Information Theory and Statistical Mechanics. *Physics Review, 106*(4), 620-630.

Jaynes, E. T. (1957). Information Theory and Statistical Mechanics II. *Physics Review, 108*(2), 171-190.

Jelinek, F. (1976). Continuous speech recognition by statistical methods. *Proceedings of the IEEE, 64*(4), 532-556.

Jelinek, F., Bahl, L., & Mercer, R. (1975). Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech. *IEEE Transactions on Information Theory, 21*(3), 250-256.

Jelinek, F., Merialdo, B., Roukos, S., & Strauss, M. I. (1990). Self organized Language Modeling for Speech Recognition. In A. Waibel, & K.-F. Lee (Eds.), *Readings in Speech Recognition* (pp. 450-506). San Mateo, CA: Morgan Kauffman Publishers, Inc.

Jensen, P. (2007). *The Behavioral Biology of Dogs.* Wallingford, UK: CABI.

Jeopardy. (2011, September 6). *IBM and 'Jeopardy!' Relive History with Encore Presentation of 'Jeopardy!: The IBM Challenge'*. Retrieved February 12, 2013, from http://www.jeopardy.com/news/watson1x7ap4.php

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Lecture Notes in Computer Science, 1398*, 137-142.

Johnson, M. (n.d.). *Incorrect Parses.* Retrieved Feb 20, 2013, from http://www.cog.brown.edu/~mj/code/incorrect-parses.txt.gz

Johnson, M., & Willsky, A. (2013). Bayesian Nonparametric Hidden Semi-Markov Models. *Journal of Machine Learning Research, 14*, 673-701.

Johnson-Laird, P. N., & Garnham, A. (1980). Descriptions and discourse models. *Linguistics and Philosophy, 3*(3), 371-393.

Jolliffe, I. (2005). *Principle Component Analysis.* New York, NY: John Wiley & Sons.

Jones, C. (2012, January 25). *If You Could Master All the Data in the World...* Retrieved August 10, 2013, from Esquire: http://www.esquire.com/features/roger-craig-jeopardy-champ-0212

Joshi, A. K., & Schabes, Y. (1997). Tree-Adjoining Grammars. In G. Rozenberg, & A. Salomaa (Eds.), *Handbook of Formal Languages 3* (pp. 69-124). New York, NY: Springer.

Joshi, A., Vijay-Shanker, K., & Weir, D. (1991). Mildly Context-Sensitive Grammars. In P. Sells, S. Shieber, & T. Wasow (Eds.), *Foundational Issues in Natural Language Processing* (pp. 31-81). Cambridge, MA: MIT Press.

Jurafsky, D., & Martin, J. H. (2000). Chapter 6: Hidden Markov and Maximum Entropy Models. In *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2nd ed., pp. 173-212). Upper Saddle River, NJ: Prentice Hall PTR.

Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Upper Saddle River, NJ: Prentice Hall PTR.

Kahneman, D. (2011). *Thinking Fast and Slow.* New York, NY: Farrar, Straus, and Giroux.

Kakkonen, T., & Sutinen, E. (2006). Towards a Framework for Evaluating Syntactic Parsers. *Lecture Notes in Computer Science, 4139*, 704-715.

Kallestinova, E. D. (2007). *Aspects of word order in Russian.* Iowa City, Iowa: The University of Iowa.

Karlsson, F., Voutilainen, A., Heikkilä, J., & Anttila, A. (1995). *Constraint Grammar: A Language-Independent System for Parsing Running Text.* Berlin, Germany: De Gruyter Mouton.

Karp, R. M. (1972). Reducibility among combinatorial problems. In R. E. Miller, & J. W. Thatcher (Eds.), *Complexity of Computer Computations* (pp. 85-103). New York, NY: Plenum.

Karttunen, L., & Beesley, K. R. (2005). Twenty-Five Years of Finite-State Morphology. In *Inquiries into Words, a Festschrift for Kimmo Koskenniemi on his 60th Birthday* (pp. 71-83). Stanford, CA: CSLI Studies in Computational Linguistics.

Kasai, T. (1970). An hierarchy between context-free and context-sensitive languages. *Journal of Computer and System Sciences, 4*(5), 492-508.

Kasami, T. (1965). *An efficient recognition and syntax-analysis algorithm for context-free languages.* Honolulu, Hawaii: Hawaii University Honolulu Deptartment of Electrical Engineering.

Katz, B., & Lin, J. (2002). Annotating the Semantic Web Using Natural Language. *Proceedings of the 2nd workshop on NLP and XML.* Taipei: Association for Computational Linguistics.

Katz, B., Lin, J., & Quan, D. (2002). Natural Language Annotations for the Semantic Web. *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE.* Irvine, CA: Springer.

Katz, S. M. (1996). Distribution of content words and phrases in text and language modelling. *Natural Language Engineering, 2*(1), 15-59.

Kaufmann, E., & Bernstein, A. (2007). How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users? *The Semantic Web: Lecture Notes in Computer Science, 4825*, 281-294.

Kautz, H., & Selman, B. (1992). Planning as Satisfiability. *Proceedings of the 10th European Conference on Artificial Intelligence (ECAI 92).* Vienna, Austria: John Wiley and Sons.

Kautz, H., & Selman, B. (1996). Pushing the Envelope: Planning, Propositional Logic, and Stochastic Search. *Proceedings of the 13th National Conference on Artificial Intelligence* (pp. 558-562). Portland, OR: AAAI Press.

Kautz, H., & Selman, B. (1999). Unifying SAT-based and Graph-based Planning. *Workshop on Logic-Based Artificial Intelligence.* Washington, D.C.: AAAI Press.

Kazama, J., Miyao, Y., & Tsujii, J. (2001). A Maximum Entropy Tagger with Unsupervised Hidden Markov Models. *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium.* Tokyo, Japan: Asian Federation of Natural Language Processing Associations.

Kazuhiro, K., Takashi, S., & Sin-ya, A. (1992). Association-based Natural Language Processing with Neural Networks. *Proceedings of the 30th Annual Meeting of the*

Association for Computational Linguistics. Newark, NJ: Association for Computational Linguistics.

Kearns, K. (2000). *Semantics.* New York, NY: St. Martin's Press.

Kearns, M. (1988). *Thoughts on Hypothesis Boosting.* (Unpublished Manuscript). University of Pennsylvania, Philadephia, PA.

Kellogg, P. (n.d.). *An Introduction to Latent Semantic Analysis.* Retrieved February 26, 2013, from http://www.patrickkellogg.com/school/papers/LSA.htm

Kelly, D., Dillingham, M., Hudson, A., & Wiesner, K. (2012). A New Method for Inferring Hidden Markov Models from Noisy Time Sequences. *PLoS One, 7*(1), e29703.

Kelly, N., & Stewart, R. G. (2010). *The Language Services Market: 2010.* Lowell, MA: Common Sense Advisory, Inc.

Kemp, J. A. (1986). The Tekhne Grammatike of Dionysius Thrax: Translated into English. *Historiographia linguistica, 13*(2-3), 2-3.

Kempe, A. (1997). Finite State Transducers Approximating Hidden Markov Models. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics.* Sofia, Bulgaria: Association for Computational Linguistics.

Kersting, K., De Raedt, L., & Raiko, T. (2006). Logical Hidden Markov Models. *Journal of Artificial Intelligence Research, 25*(1), 425-456.

Kim, H., Howland, P., & H., P. (2006). Dimension Reduction in Text Classification with Support Vector Machines. *Journal of Machine Learning Research, 6*(1), 37-56.

Kim, S., & Zhang, H. (1994). ModGen: Theorem proving by model generation. *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)* (pp. 162-167). Seattle, WA: AAAI Press.

Kindermann, R., & Snell, J. L. (1980). *Markov Random Fields and Their Applications.* Providence, RI: American Mathematics Society.

Kirby, S. (2007). The Evolution of Language. In *Oxford handbook of evolutionary psychology* (pp. 669-681). Oxford, UK: Oxford University Press.

Kish, L. B. (2002). End of Moore's law: thermal (noise) death of integration in micro and nano electronics. *Physics Letters A, 305*(3), 144-149.

Klein, D., & Manning, C. D. (2001). Parsing with Treebank Grammars: Empirical Bounds, Theoretical Models, and the Structure of the Penn Treebank. *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics.* Toulouse, France: Association for Computational Linguistics.

Klein, D., & Manning, C. D. (2003). Accurate unlexicalized parsing. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (Volume 1).* Morristown, NJ: Association for Computational Linguistics.

Knuth, D. E. (1965). On the translation of languages from left to right. *Information and Control, 8*(6), 607-639.

Kolodner, J. L. (1992). An Introduction to Case-Based Reasoning. *Artificial Intelligence Review, 6*(1), 3-34.

Kolya, A. K., Ekbal, A., & Bandyopadhyay, S. (2010). Event-event relation identification: A CRF based approach. *Proceedings of the Sixth International Conference on Natural Language Processing and Knowledge Engineering.* Beijing, China: IEEE Press.

Krough, A., Brown, M., Mian, I., Sjölander, K., & Haussler, D. (1994). Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology, 235*(5), 1501-1531.

Kubler, S., & Prokic, J. (2006). Why is German Dependency Parsing More Reliable than Constituent Parsing? *Proceedings of the 5th Work on Treebanks and Linguistic Theories (TLT).* Prague, Czech Republic: Institute of Formal and Applied Linguistics.

Kubler, S., Maier, W., Rehbein, I., & Versley, Y. (2008). How to Compare Treebanks. *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08).* Marrakech, Morocco: European Language Resources Association (ELRA).

Kudova, P. (2006). *Learning with Regularization Networks*. (Doctoral Dissertation). Charles University, Prague, Czech Republic.

Kuhn, T. S. (2012). *The structure of scientific revolutions.* Chicago, IL: University of Chicago Press.

Kurimo, M., Puurula, A., Arisoy, E., Siivola, V., Hirsimäki, T., Pylkkönen, J., . . . Saraclar, M. (2006). Unlimited vocabulary speech recognition for agglutinative languages. *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics.* New York, NY: Association of Computational Linguistics.

Kuroda, S. Y. (1964). Classes of languages and linear-bounded automata. *Information Control, 7*(2), 207-223.

Kurup, U., & Cassimatis, N. L. (2010). Integrating Constraint Satisfaction and Spatial Reasoning. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)* (pp. 1536-1541). Atlanta, GA: AAAI Press.

Kurup, U., & Cassimatis, N. L. (2010). Quantitative Spatial Reasoning for General Intelligence. *Proceedings of AGI-10* (pp. 1-6). Lugano, Switzerland: Atlantis Press.

Kwan, C., Yin, J., Ayhan, B., Chu, S., Puckett, K., Zhao, Y., . . . Sityar, I. (2008). Speech Separation Algorithms for Multiple Speaker Environments. *Proceedings of the 5th International Symposium on Neural Networks.* Beijing, China: Springer.

La, L., Guo, Q., & Cao, Q. (2012). Training MEMM with PSO: A Tool for Part-of-Speech Tagging. *Journal of Software, 7*(11), 2511-2517.

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of the 18th International Conference on Machine Learning.* Williamstown, MA: Morgan Kaufmann Publishers.

Lakoff, G. (1993). The Contemporary Theory of Metaphor. In A. Ortony (Ed.), *Metaphor and Thought* (2nd ed., pp. 202-251). Cambridge, UK: Cambridge University Press.

Lakoff, G., & Johnson, M. (1980). *Metaphors We Live By.* Thousand Oaks, CA: Pine Forge Press.

Lakoff, G., & Johnson, M. (2008). *Women, Fire, and Dangerous Things.* Chicago, IL: University of Chicago Press.

Lamel, L. F., & Gauvain, J. L. (1991). BREF, a Large Vocabulary Spoken Corpus for French. *Proceedings of EUROSPEECH-91.* Genova, Italy: European Speech Communication Association.

Lamel, L. F., & Gauvain, J. L. (1993). High Performance Speaker-Independent Phone Recognition Using CDHMM. *EUROSPEECH 93* (pp. 121-124). Berlin, Germany: ISCA Archive.

Land, A. H., & Doig, A. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica, 28*(3), 497-520.

Landauer, T. K., Foltz, P. W., & Latham, D. (1998). An Introduction to Latent Semantic Analysis. *Discourse Processes, 25*(2-3), 259-284.

Langley, P., & Choi, D. (1999). A Unified Cognitive Architecture for Physical Agents. *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99). 21.* Menlo Park, CA: AAAI Press.

Langley, P., Choi, D., & Rogers, S. (2005). *Interleaving Learning, Problem Solving, and Execution in the Icarus Architecture.* Stanford, CA: Stanford University Center for the Study of Language and Information.

Langley, P., Laird, J. E., & Rogers, S. (2009). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research, 10*(2), 141-160.

Lappin, S. (1995). The interpretation of ellipsis. In S. Lappin (Ed.), *The Handbook of Contemporary Semantic Theory* (pp. 123-145). Cambridge, MA: Blackwell.

Lappin, S. (1996). The interpretation of ellipsis. In S. Lappin (Ed.), *Handbook of contemporary semantic theory* (pp. 145-175). New York, NY: Blackwell.

Lauritzen, S. L. (1996). *Graphical Models (Volume 17).* New York, NY: Oxford University Press.

Law, V., & Sluiter, I. (1995). *Dionysius Thrax and the Technē grammatikē.* Münster, Germany: Nodus Publikationen.

Leacock, C., & Chodorow, M. (1998). Combining Local Context and WordNet Similarity for Word Sense Identification. *WordNet: An Electronic Lexical Database, 49*(2), 265-283.

Lee, D. (2010, March 11). *Non-English, Parallel & Multilingual Corpora (a selection)*. (University of Wollongong) Retrieved February 10, 2013, from http://www.uow.edu.au/~dlee/corpora2.htm

Legg, S., & Hutter, M. (2007). A Collection of Definitions of Intelligence. In *Advances in Artificial General Intelligence: Concepts, Architectures, and Algorithms* (Vol. 157, pp. 17-24). Amsterdam, Netherlands: IOS Press.

Leibniz, G. W., & Alexander, H. G. (1998). *The Leibniz-Clarke correspondence.* New York, NY: Manchester University Press.

Leidner, J. L. (2003). Current Issues in Software Engineering for Natural Language Processing. *Proceedings of the HLT-NAACL 2003 Workshop on Software Engineering and Architecture of Language Technology Systems (SEALTS '03).* Edmonton, Canada: Association for Computational Linguistic.

Lember, J. (2011). On approximation of smoothing probabilities for hidden Markov models. *Statistics & Probability Letters, 81*(2), 310-316.

Lenat, D. B., & Guha, R. V. (1993). *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project.* Boston, MA: Addison-Wesley.

Lewis, D. D., & Jones, K. S. (1996). Natural Language Processing for Information Retrieval. *Communications of the ACM, 39*(1), 92-101.

Li, B., Shiwen, Y., & Lu, Q. (2012). An improved k-Nearest Neighbor Algorithm for Text Categorization. *Expert Systems with Applications: An International Journal, 39*(1), 1503-1509.

Li, J., & Larsen, K. (2011). Establishing Nomological Networks for Behavioral Science: a Natural Language Processing Based Approach. *Proceedings of the 2011 International Conference on Information Systems.* Shanghai, China: ICIS.

Li, P., Farkas, I., & MacWhinney. (2004). Early lexical development in a self-organizing neural network. *Neural Networks, 17*(8-9), 1345-1362.

Li, X., Wang, X., & Yao, L. (2011). Joint Decoding for Chinese Word Segmentation and POS Tagging Using Character-Based and Word-Based Discriminative Models. *Proceedings of the 2011 International Conference on Asian Language Processing (IALP).* Penang, Malaysia: Chinese and Oriental Languages Information Processing Society.

Li, Y., Duan, H., & Zhai, C. (2012). A Generalized Hidden Markov Model with Discriminative Training for Query Spelling Correction. *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval.* Portland, OR: Association for Computational Machinery.

Lifestyle Lounge. (n.d.). *Euphemism Examples.* Retrieved February 27, 2013, from http://lifestyle.iloveindia.com/lounge/euphemism-examples-12691.html

Liggett, T. M. (2010). Continuous Time Markov Processes: An Introduction. *Graduate Studies in Mathematics, 113*.

Lin, Y., Michel, J.-B., Aiden, E. L., Orwant, J., Brockman, W., & Petrov, S. (2012). Syntactic Annotations for the Google Books Ngram Corpus. *Proceedings of the ACL 2012 System Demonstrations.* Jeju Island, Korea: Association for Computational Linguistics.

Lita, L. V., Ittycheriah, A., Rouk, S., & Kambhatla, N. (2003). tRuEcasing. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics.* Sapporo, Japan: Association for Computational Linguistics.

Liu, Y., & Levesque, H. J. (2005). Tractable Reasoning in First-Order Knowledge Bases with Disjunctive Information. *Proceedings of the National Conference on Artificial Intelligence (Vol. 20 No. 2).* Menlo Park, CA: AAAI Press.

Liu, Y., Shriberg, E., Stolcke, A., & Harper, M. (2005). Comparing HMM, Maximum Entropy, and Conditional Random Fields for Disfluency Detection. *Proceeedings of the 9th European Conference on Speech Communication and Technology.* Lisbon: Interspeech.

Loebner, H. (1993). *In Response.* Retrieved December 13, 2012, from http://loebner.net/Prizef/In-response.html

Luber, S. (2011). Cognitive Science Artificial Intelligence: Simulating the Human Mind to Achieve Goals. *IEEE 3rd International Conference on Computer Research and Development (ICCRD) (Volume 1).* Shanghai, China: IEEE Press.

Ludeling, A., & Kyto, M. (Eds.). (2008). *Corpus Linguistics: An International Handbook.* Berlin, Germany: De Gruyter Mouton.

MacCartney, B. (2009). *Natural Language Inference.* Stanford, CA: Stanford University.

MacKenzie, I. S., & Soukoreff, R. W. (2002). Text entry for mobile computing: Models and Methods, theory and practice. *Human-Computer Interaction, 2-3*(17), 147-198.

MacKenzie, I. S., & Tanaka-Ishii, K. (2007). *Text Entry Systems: Mobility, Accessibility, Universality.* San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Makhoul, J., Kubala, F., Schwartz, R., & Weischedel, R. (1999). Performance Measures for Information Extraction. *Proceedings of the DARPA Broadcast News Workshop* (pp. 249-252). Lansdowne, VA: DARPA.

Malode, A. A., & Sahare, S. (1963). Advanced Speaker Recognition. *International Journal of Advances in Engineering and Technology, 4*(1), 443-455.

Manjoo, F. (2012, November 15). *Siri Is a Gimmick and a Tease*. (Slate) Retrieved March 10, 2013, from http://www.slate.com/articles/technology/technology/2012/11/siri_vs_google_the_search_company_s_voice_recognition_program_gets_closer.html

Manning, C. (2011). Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics? *Proceedings of the 12th international Conference on Computational Linguistics and Intelligent Text Processing (CICLing'11)* (pp. 171-189). Tokyo, Japan: Springer-Verlag Berlin.

Manning, C. D., & Schutze, H. (1999). Chapter 12: Probabilistic Parsing. In *Foundations of Statistical Natural Language Processing* (pp. 407-460). Cambridge, MA: MIT Press.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing.* Cambridge, MA: MIT Press.

Marathe, M. V., Hunt III, H. B., & Ravi, S. S. (1993). *The Complexity of Approximating PSPACE-Complete Problems for Hierarchical Specifications.* Berlin, Germany: Springer Berlin Heidelberg.

Marcus, G. (2008). *Kludge: The Haphazard Evolution of the Human Mind.* New York, NY: Houghton Mifflin Harcourt.

Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. (1993). Buidling a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics, 19*(2), 313-330.

Markman, E. M. (1994). Constraints on word meaning in early language acquisition. *Lingua, 92*, 199-227.

Markov, A. (1913). An Example of Statistical Analysis of the Text of Eugene Onegin Illustrating the Association of Trials into a Chain. *Bulletin de l'Academie Imperiale des Sciences de St. Petersburg, 6*(7), 153-162.

Markov, A. (1971). Extension of the limit theorems of probability theory to a sum of variables connected in a chain. In *Dynamic Probabilistic System (Volume 1): Markov Chains* (p. Appendix B). Hoboken, NJ: John Wiley and Sons.

Marques-Silva, J. P., & Sakallah, K. A. (2000). Boolean satisfiability in electronic design automation. *Proceedings of the IEEE/ACM Design Automation Conference (DAC '00).* Washington, DC: Association for Computing Machinery.

Marr, D. (1982). *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information.* New York, NY: Henry Holt and Co., Inc.

Marshall, M. (2008, June 26). *Microsoft to buy semantic search engine Powerset for $100M plus.* Retrieved December 20, 2012, from http://venturebeat.com/2008/06/26/microsoft-to-buy-semantic-search-engine-powerset-for-100m-plus/

Matilal, B. K. (1990). *The word and the world: India's contribution to the study of language.* Delhi, India: Oxford University Press.

Mauldin, M. L. (1994). Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition. *Proceedings of the National Conference on Artifical Intelligence.* Hoboken, NJ: John Wiley & Sons LTD.

May, M. (1998). Diagrammatic reasoning and levels of schematization. In M. Skov (Ed.), *Iconicity.* Copenhagen, Denmark: NSU Press.

Maybury, M. T., & Wahlster, W. (1998). *Readings in intelligent user interfaces.* Burlington, MA: Morgan Kaufmann.

McCallum, A., Freitag, D., & Pereira, F. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. *Proceedings of the 17th International Conference on Machine Learning (Volume 951 Issue 008)* (pp. 591-598). Stanford, CA: ICML.

McCarthy, J. (1980). Circumscription – A form of non-monotonic reasoning. *Artificial Intelligence, 13*(1), 27-39.

McCarthy, J. (1987). Generality in artificial intelligence. *Communications of the ACM, 30*(12), 1030-1035.

McCarthy, J. (2007). From Here to Human-Level AI. *Artificial Intelligence, 171*(18), 1174-1182.

McClelland, J. L. (2009). Connectionist models. In T. Byrne, & A. Cleeremans (Eds.), *Oxford Companion to Consciousness* (pp. 177-181). New York, NY: Oxford University Press.

McClelland, J. L., & Elman, J. L. (1986). The TRACE model of speech perception. *Cognitive Psychology, 18*, 1-86.

McClosky, D., Charniak, E., & Johnson, M. (2006). Effective Self-Training for Parsing. *Proceedings of the Main Conference of the North American Chapter of the Association of Computational Linguistics.* New York, NY: Association for Computational Linguistics.

McShane, M. (2009). Reference Resolution Challenges for Intelligent Agents: The Need for Knowledge. *IEEE Intelligent Systems, 24*(4), 47-58.

Medlock, B. (2008). *Investigating classification for natural language processing tasks.* Saarbrücken, Germany: VDM Publishing.

Menard, O., & Frezza-Buet, H. (2005). Model of multi-modal cortical processing: Coherent learning in self-organizing modules. *Neural Networks, 18*(5), 646-655.

Metcalfe, J., & Shimamura, A. P. (1994). *Metacognition: knowing about knowing.* Cambridge, MA: MIT Press.

Meteyard, L., & Vigliocco, G. (2008). The Role of Sensory and Motor Information in Semantic Representation - A Review. In *Handbook of Cognitive Science: An Embodied Approach.* London, England: Academic Press, Elsevier.

Metropolis, N., & Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association, 44*(247), 335-341.

Meyer, D. E., & Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance. *Psychological review, 104*(1), 3-65.

Michel, J. B.-K., Aiden, A. P., Veres, A., Gray, M. K., Team, T. G., ..., & Aiden, E. L. (2011). Quantitative Analysis of Culture Using Millions of Digitized Books. *Science, 331*(6014), 176-182.

Miller, G. A., Beckwith, R., Fellbaum, C. D., Gross, D., & Miller, K. (1990). WordNet: An online lexical database. *International Journal of Lexicography, 3*(4), 235-312.

Miller, L. A. (1981). Natural language programming: Styles, strategies, and contrasts. *IBM Systems Journal, 20*(2), 184-215.

Milnes, B. G., Pelton, G., Doorenbos, R., Laird, M. H., Rosenbloom, P., & Newell, A. (1992). *A Specification of the Soar Cognitive Architecture in Z: Technical Report CMU-CS-92-169.* Pittsburgh, PA: Carnegie Mellon University School of Computer Science.

Minsky, M. (1975). A Framework for Representing Knowledge. In P. H. Winston (Ed.), *The Psychology of Computer Vision* (pp. 211-277). New York, NY: McGraw-Hill.

Minsky, M., & Papert, S. (1969). *Perceptrons.* Cambridge, MA: MIT Press.

Mohri, M. (1997). Finite-State Transducers in Language and Speech Processing. *Computational Linguistics, 23*(2), 269-311.

Molina, A., Pia, F., & Segarra, E. (2002). A Hidden Markov Approach to Word Sense Disambiguation. In *Advances in Artificial Intelligence (IBERAMIA 2002)* (pp. 655-663). Berlin, Germany: Springer Berlin Heidelberg.

Moore, G. E. (1975). Progress in digital integrated electronics. *1975 International Electron Devices Meeting, 21*, 11-13.

Morris, J. B. (1969). E-resolution: extension of resolution to include the equality relation. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI-69)* (pp. 287-294). Washington D.C.: International Joint Conferences on Artificial Intelligence.

Moskewicz, M., Madigan, C. F., Zhao, Y., Zhang, L., & Malik, S. (2001). Chaff: Engineering an Efficient SAT Solver. *Proceedings of the 39th Design Automation Conference.* Las Vegas, NV: DAC.

Moura, M. C., & Droguett, E. L. (2008). A continuous-time semi-markov bayesian belief network model for availability measure estimation of fault tolerant systems. *Pesquisa Operacional, 28*(2), 355-375.

Mueller, E. T. (2004). Event calculus reasoning through satisfiability. *Journal of Logic and Computation, 14*(5), 703-730.

Murphy, K. P. (2002). Dynamic Bayesian Networks. In M. Jordan (Ed.), *Probabilistic Graphical Models.* Cambridge, MA: MIT Press.

Murphy, S. (2012). *Design Considerations for a Natural User Interface (NUI).* Dallas, TX: Texas Instruments Inc.

Murty, K. G. (1983). *Linear programming (Volume 57).* New York, NY: Wiley.

Murugesan, A. (2009). *A framework for representing and jointly reasoning over linguistic and non-linguistic knowledge*. (Doctoral Dissertation). Rensselaer Polytechnic Institute, Troy, NY.

Mystre, S. M., Savova, G. K., Kipper-Schuler, K. C., & Hurdle, J. F. (2008). Extracting information from textual documents in the electronic health record: a review of recent research. *Yearbook of Medical Informatics, 47*(Suppplement 1), 128-144.

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes, 30*(1), 3-26.

Narayanan, S., & Jurafsky, D. (1998). A Bayesian Model of Human Sentence Processing. *Proceedings of the 12th Annual Meeting of the Cognitive Science Society.* Boston, MA: Lawrence Erlbaum Associates.

Nash, D. G. (1980). *Topics in Warlpiri Grammar.* Cambridge, MA: Massachusetts Institute of Technology.

Natarajan, S., Bui, H. H., Tadepalli, P., Kersting, K., & Wong, W.-K. (2008). *Logical Hierarchical Hidden Markov Models for Modeling User Activities.* Berlin, Germany: Springer Berlin Heidelberg.

Nenkova, A., & McKeown, K. (2011). *Automatic Summarization.* Hanover, MA: Now Publishers, Inc.

Nerlove, M., & Press, S. J. (1973). *Univariate and Multivariate Log-Linear and Logistic Models. Vol. 1306.* Santa Monica, CA: Rand.

Newborn, M., & Newborn, M. (1997). *Kasparov vs. Deep Blue: Computer chess comes of age.* New York, NY: Springer-Verlag New York, Inc.

Newell, A. (1973). *You Can't Play 20 Questions with Nature and Win: Projective Comments on the Papers of this Symposium.* Pittsburgh, PA: Carnegie Mellon University, Deptartment of Computer Science.

Newell, A. (1994). *Unified Theories of Cognition.* Cambridge, MA: Harvard University Press.

Newell, A., & Simon, H. (1956). The Logic Theory Machine - A Complex Information Processing System. *IRE Transactions on Information Theory, 2*(3), 61-79.

Newell, A., & Simon, H. (1976). Computer Science as Empirical Inquiry - Symbols and Search. *Communications of the ACM, 19*(3), 113-126.

Newell, A., Shaw, J. C., & Simon, H. A. (1959). *Report on a general problem-solving program.* Santa Monica, CA: Rand Corporation.

Ng, H. T., & Low, J. K. (2004). Chinese Part-of-Speech Tagging: One-at-a-Time or All-at-Once? Word-Based or Character-Based? *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP Volume 4).* Barcelona, Spain: Association for Computational Linguistics.

Nicholas, N., & Cowan, J. W. (2003). *What Is Lojban?* San Mateo, CA: Logical Language Group.

Nicuagain. (2011, November 1). *Siri User Guide: Full List of Siri Commands*. Retrieved May 22, 2013, from Idoono: http://www.idoono.com/2011/11/siri-user-guide-siri-commands/

Nirenburg, S., McShane, M., Zabludowski, M., Beale, S., & Pfeifer, C. (2005). *Ontological Semantic Text Processing in the Biomedical Domain.* Baltimore, MD: University of Maryland Baltimore County, Institute for Language and Information Technologies.

Nivre, J. (2005). *Dependency Grammar and Dependency Parsing.* Växjö, Sweden: Vaxjo University: School of Mathematics and Systems Engineering.

Norman, D. (2002). *The Design of Everyday Things.* New York, NY: Basic Books.

Norman, D., & Dunaeff, T. (1994). *Things that Make Us Smart: Defending Human Attributes in the Age of the Machine.* New York, NY: Basic Books.

Norvig, P. (2011). *On Chomsky and the Two Cultures of Statistical Learning.* Retrieved December 13, 2012, from http://norvig.com/chomsky.html

Norvig, P. (2011, February 12). The Machine Age. *New York Post.* Retrieved April 21, 2013 from http://nypost.com/2011/02/13/the-machine-age/

Noureddine, S. (2005). An Approach for the Satisfiability Problem via Exterior Penalty Optimization. *Journal of Computer Science, Special Issue*.

NPR Staff. (2011, November 20). *How One Man Played 'Moneyball' With 'Jeopardy!'.* Retrieved August 3, 2013, from NPR: http://www.npr.org/2011/11/20/142569472/how-one-man-played-moneyball-with-jeopardy

Nuance Communications, Inc. (2007, June 21). *Nuance to Acquire Tegic Communications.* Retrieved December 13, 2012, from http://www.nuance.com/news/pressreleases/2007/20070621_tegic.asp

Nuance Communications, Inc. (2012, August 07). *Nuance Announces Third Quarter Fiscal 2012 Results.* Retrieved Dececember 13, 2012, from http://www.nuance.com/company/news-room/press-releases/2012-08-07-Nuance-Q3-2012-Press-Release.doc

Nusca, A. (2011, November 3). *How Apple's Siri really works*. (ZDNet) Retrieved March 10, 2013, from http://www.zdnet.com/blog/btl/how-apples-siri-really-works/62461

Oaks, D. D. (2012). On a Possible Generative Approach to Structurally Ambiguous Humor. *2012 AAAI Fall Symposium Series.* Arlington, VA: AAAI Press.

Oard, D. W., & Diekema, A. R. (1998). Cross-Language Information Retrieval. *Annual Review of Information Science, 33*, 223-256.

Obleser, J., & Kotz, S. A. (2011). Multiple brain signatures of integration in the comprehension of degraded speech. *Neuroimage, 55*(2), 713-723.

O'Hagan, A., Forster, J., & Kendall, M. G. (2004). *Bayesian inference.* London, UK: Arnold.

Okasa, S. (2003). Fodor on Cognition, Modularity, and Adaptationism. *Philosophy of Science, 70*, 68-88.

O'Neill, B., Barlow, T. M., Safranek, D., & Beige, A. (2012). Hidden Quantum Markov Models with one qubit. *American Institute of Physics Conference Series, 1479*, 667-771.

OpenCog. (2012, September 12). *Natural Language Processing*. (OpenCog) Retrieved January 4, 2013, from http://wiki.opencog.org/w/Natural_Language_Processing

Ord, T. (2002). *Hypercomputation: Computing more than the Turing machine.* Melbourne, Australia: University of Melbourne Department of Philosophy.

Ormrod, J. E., & Davis, K. M. (2004). *Human learning.* London, England: Merrilll.

Ornstein, R. E. (1997). *The Right Brain: Making Sense of the Hemispheres.* New York, NY: Harcourt Brace.

Orphanos, G., Kalles, D., & Papagelis, A. (1999). Decision Trees and NLP: A Case Study in POS Tagging. *Proceedings of Annual Conference on Artificial Intelligence (ACAI).* Chania, Crete: International Conference on Automatic Control and Artificial Intelligence.

Oxford Dictionaries. (2013). *How many words are there in the English language.* Retrieved April 1, 2013, from http://oxforddictionaries.com/words/how-many-words-are-there-in-the-english-language

Oxford Dictionaries. (2013). *Is it true that English has the most words of any language.* Retrieved April 1, 2013, from http://oxforddictionaries.com/words/is-it-true-that-english-has-the-most-words-of-any-language

Pak, A., & Paroubek, P. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010).* Valletta, Malta: European Language Resources Association.

Palmer, M., Gildea, D., & Xue, N. (2010). Semantic role labeling. *Synthesis Lectures on Human Language Technologies, 3*(1), 1-103.

Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval, 2*(1-2), 1-135.

Panini. (1987). *The Astadhyayi of Panini.* Austin, TX: University of Texas Press.

Pantel, P., & Pennacchiotti, M. (2006). Espresso: leveraging generic patterns for automatically harvesting semantic relations. *Proceedings of the 21st International*

Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics. Sydney, Australia: Association for Computational Linguistics.

Papageorgiou, C. P. (1994). Japanese Word Segmentation by Hidden Markov Model. *Proceedings of the Workshop on Human Language Technologies (HLT '94).* Morristown, NJ: Association for Computational Linguistics.

Papanicolaou, A. (2011). *The Baum-Welch & Viterbi Algorithms.* Princeton, NJ: Princeton University Stocastic Analysis Seminar.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02).* Philadelphia, PA: Association for Computational Linguistics.

Pasula, H., Marthi, B., Milch, B., Russell, S., & Shpitser, I. (2002). Identity uncertainty and citation matching. *Advances in Neural Information Processing (NIPS 2002), 15*, 1401-1408.

Patrick, J. (2010, February 12). *Medical NLP and Engineering. An NLP Workbench for it.* Universidad del Pais Vasco, Basque Region, Spain.

Paul, D. B., & Baker, J. M. (1992). The Design for the Wall Street Journal-based CSR Corpus (HLT '91). *Proceedings of the Workshop on Speech and Natural Language.* Morristown, NJ: Association for Computational Linguistics.

Peng, F. (2001). *Automatic Multi-Lingual Information Extraction.* Waterloo, Canada: University of Waterloo, Canada.

Penniston, A., & Harley, E. (2011). Attempts to Verify Written English. *Proceedings of C* Conference on Computer Science and Software Engineering 2011.* Montreal, Canada: ACM.

Pereira, F. C., & Shieber, S. M. (2002). *Prolog and Natural-Language Analysis.* Brookline, MA: Microtome Publishing.

Perlovsky, L. I. (2010). The Mind is not a Kludge. *Skeptic, 15*(3), 51-55.

Petrovic, S., Osborne, M., & Lavrenko, V. (2010). The Edinburgh Twitter Corpus. *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in*

a World of Social Media. Los Angeles, CA: Association for Computational Linguistics.

Pineda, F. J. (1987). Generalization of back-propagation to recurrent neural networks. *Physics Review Letters, 59*(19), 2229-2232.

Pinker, S. (1994). *The Language Instinct.* New York, NY: Harper Perennial Modern Classics.

Pinker, S., & Jackendoff, R. (2005). The faculty of language: What's special about it? *Cognition, 95*(2), 201-236.

Pisoni, D., & Remez, R. (Eds.). (2005). *The Handbook of Speech Perception.* Oxford, UK: Blackwell.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers, 10*(3), 61-74.

Plunkett, K., & Marchman, V. (1993). From rote learning to system building: Acquiring verb morphology in children and connectionist nets. *Cognition, 48*(1), 21-69.

Pollard, C. A., & I., S. (1994). *Head-driven phrase structure grammar.* Chicago, IL: University of Chicago Press.

Pollette, C. (n.d.). *How Wolfram|Alpha Works*. (How Stuff Works) Retrieved March 10, 2013, from http://computer.howstuffworks.com/internet/basics/wolfram-alpha.htm

Poon, H. (2011). *Markov Logic for Machine Reading.* Seattle, WA: University of Washington.

Poon, H., & Domingos, P. (2009). Unsupervised Semantic Parsing. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. 1.* Singapore: Association for Computational Linguistics.

Poon, H., & Domingos, P. (2011). Sum-Product Networks - A New Deep Architecture. *Proceedings of the IEEE International Conference on Computer Vision Workshop (ICCV Workshop).* Barcelona, Spain: IEEE Press.

Poon, H., Domingos, P., & Sumner, M. (2008). A general method for reducing the complexity of relational inference and its application to MCMC. *Proceedings of AAAI-08.* Chicago, IL: AAAI Press.

Popescu, A.-M., Pennacchiotti, M., & Paranjpe, D. (2011). Extracting Events and Event Descriptions from Twitter. *Proceedings of the 20th International Conference Companion on World Wide Web.* Hyderabad, India: ACM.

Potapov, A., Rodionov, S., Myasnikov, A., & Begimov, G. (2005). Cognitive Bias for Universal Algorithmic Intelligence. *Proceedings of the 5th International Conference on Artificial General Intelligence (AGI-12).* Oxford, England: AGI.

Powers, D. M. (2008). "Evaluation Evaluation" powers. *Proceedings of the 18th European Conference on Artificial Intelligence.* Patras, Greece: IOS Press.

Powers, D. M. (2008). Evaluation Evaluation-a Monte Carlo Study. *Proceedings of the 18th European Conference on Artificial Intelligence.* Patras, Greece: IOS Press.

Prolo, C. A. (2002). Fast LR Parsing Using Rich Tree Adjoining Grammars. *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing (Volume 10).* Philadelphia, PA: Association for Computational Linguistics.

Prosser, P. (1993). Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence, 9*(3), 268-299.

Protege. (2009, November 30). *Protege-OWL Reasoning API.* Retrieved March 10, 2013, from http://protegewiki.stanford.edu/wiki/ProtegeReasonerAPI

Przybyla, J., & Zhou, X. (2008, December). *Cell Phone Use While Driving: A Literature Review and Recommendations.* Retrieved December 13, 2013, from http://www.key2safedriving.net

Purver, M. (2011). Topic Segmentation. In G. Tur, & R. de Mori (Eds.), *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech* (pp. 291-317). Hoboken, NJ: Wiley.

Purves, D., Augustine, G. J., Fitzpatrick, D., Katz, L. C., LaMantia, A.-S., McNamara, J. O., & Williams, S. M. (2001). *The Development of Language: A Critical Period in Humans.* Sunderland, MA: Sinauer Associates.

Quarteroni, S., & Manandhar, S. (2009). Designing an interactive open-domain question answering system. *Natural Language Engineering, 15*(1), 73-95.

Quine, W. V. (1950). Identity, ostension, and hypostasis. *The Journal of Philosophy, 47*(22), 621-633.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning, 1*(1), 81-106.

Quinn, A. (1993). *60 Ways to Turn a Phrase.* London, UK: Routledge.

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Processing. *Proceedings of the IEEE, 77(2)*, 257-286.

Rajagopalan, B., Lall, U., & Tarboton, D. G. (1996). Nonhomogenous Markov Model for Daily Precipitation. *Journal of Hydrologic Engineering, 1*(1), 33-40.

Ramanujam, S., & Baldridge, J. (2009). Supertagging with Factorial Hidden Markov Models. *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation.* Hong Kong: City University of Hong Kong.

Ratnaparkhi, A. (1996). A Maximum Entropy Model for Part-of-Speech Tagging. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (Volume 1).* Sydney, Australia: Association for Computational Linguistics.

Ratnaparkhi, A. (1997). A Linear Observed Time Statistical Parser Based on Maximum Entropy Models. *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing EMNLP-2.* Providence, RI: Association for Computational Linguistics.

Ratnaparkhi, A. (1997). *A Simple Introduction to Maximum Entropy Models for Natural Language Processing* (Vol. 81). Philadelphia, PA: University of Pennsylvania.

Ratnaparkhi, A. (1998). *Maximum Entropy Models for Natural Language Ambiguity Resolution.* Philadelphia, PA: University of Pennsylvania.

Ratnaparkhi, A. (1998). Unsupervised Statistical Models for Prepositional Phrase Attachment. *Proceedings of the 17th International Conference on Computational Linguistics.* Montreal, Canada: Association for Computational Linguistics.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning, 34*(1-3), 151-175.

Ratnaparkhi, A., & Reynar, J. A. (1994). A Maximum Entropy Model for Prepositional Phrase Attachment. *Proceedings of the Workshop on Human Language Technology.* Plainsboro, NJ: Association for Computational Linguistics.

Ravi, S., & Knight, K. (2009). Minimized Models for Unsupervised Part-of-Speech Tagging. *Proceedings of the Joint Conference of the 47th Annual Meeting of the*

ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (Volume 1). Singapore: Association for Computational Linguistics.

Rayner, K. (1995). Eye Movements and Cognitive Processes in Reading, Visual Search, and Scene Perception. *Studies in Visual Information Processing, 6*, 3-22.

Rehbein, I., & van Genabith, J. (2007). Evaluating evaluation measures. *Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA).* Odense, Denmark: NODALIA.

Reiter, E., & Dale, R. (1997). Building Applied Natural Language Generation Systems. *Natural Language Engineering, 3*(1), 57-87.

Rennie, J. (2003). *The Use of Transfer in Natural Language Processing*. (Unpublished Manuscript). Massachusetts Institute of Technology, Cambridge, MA.

Resnik, P. (1992). Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. *Proceedings of the 14th Conference on Computational Linguistics (Volume 2).* Morristown, NJ: Association for Computational Linguistics.

Reynar, J. C., & Ratnaparkhi, A. (1997). A Maximum Entropy Approach to Identifying Sentence Boundaries. *Proceedings of the 5th Conference on Applied Natural Language Processing.* Washington, DC: Association for Computational Linguistics.

Richards, I. A., & Gibson, C. M. (1945). *Learning Basic English: A Practical Handbook for English-Speaking People.* New York, NY: WW Norton, Inc.

Richardson, M., & Domingos, P. (2006). Markov Logic Networks. *Machine Learning, 62*(1-2), 107-136.

Rissland, E. L. (1983). Examples in legal reasoning: legal hypotheticals. *Proceedings of the International Joint Conference on Artificial Intelligence.* Karlsruhe, West Germany: International Joint Conferences on Artificial Intelligence.

Rizzolatti, G. (2005). The mirror neuron system and its function in humans. *Anatomy and Embryology, 210*(5-6), 419-421.

Roche, E., & Schabes, Y. (1995). Deterministic Part-Of-Speech Tagging with Finite State Transducers. *Computational Linguistics, 21*(1), 227-253.

Rodriguez, C. G. (2010). *Parsing Schemata for Practical Text Analysis.* Singapore: World Scientific.

Roger, B. (n.d.). *Forward Algorithm.* Retrieved February 08, 2013, from http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/forward_algorithm/s1_pg1.html

Rojas, R. (1996). *Neural Networks: A Systematic Introduction.* Berlin, Germany: Springer.

Rose, M. (2012, July 23). *Siri celebrity ad series continues with Martin Scorsese*. (Tuaw) Retrieved March 10, 2013, from http://www.tuaw.com/2012/07/23/siri-celebrity-ad-series-continues-with-martin-scorsese/

Rosenberg, D., Klein, D., & Taskar, B. (2007). *Mixture-of-Parents Maximum Entropy Markov Models.* Corvallis, OR: AUAI Press.

Rosenblatt, F. (1957). *The Perceptron--a perceiving and recognizing automaton.* Ithaca, NY: Cornell Aeronautical Laboratory.

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review, 65*(6), 386-408.

Rosenbloom, P. S., Laird, J. E., & Newell, A. (1993). *The Soar Papers: Readings on Integrated Intelligence.* Cambridge, MA: MIT Press.

Rozenberg, G., & Salomaa, A. (1997). *Handbook of formal languages: Word, language, grammar (Vol. 3)*. Berlin, Germany: Springer Verlag.

Rumelhart, D. E., & McClelland, J. L. (1986). PDP models and general issues in cognitive science. In *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 1: Foundations.* Cambridge, MA: Bradford Books/MIT Press.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533-536.

Rus, V., McCarthy, P. M., Lintean, M. C., McNamara, D. S., & Graesser, A. C. (2008). Paraphrase identification with lexico-syntactic graph subsumption. *Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference.* Coconut Grove, FL: FLAIRS.

Russell, S., & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach.* Upper Saddle River, NJ: Prentice Hall.

Sacks, O. W. (1995). *An Anthropologist on Mars.* London, England: Picador.

Sag, I., Wasow, T., & Bender, E. (2000). Syntactic Theory: A Formal Introduction. *Computational Linguistics, 26*(2), 295-295.

Salakhutdinov, R. (2009). *Learning Deep Generative Models.* Toronto, Canada: University of Toronto.

Salakhutdinov, R., & Larochelle, H. (2010). Efficient Learning of Deep Boltzmann Machines. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics.* Sardinia, Italy: AIStats.

Sampson, G. (2000). A proposal for improving the measurement of parse accuracy. *International Journal of Corpus Linguistics, 5*(1), 53-68.

Sampson, G., & Babarczy, A. (2006). *Definitional and human constraints on parsing performance.* Osnabruck, Germany: Quantitative Investigations in Theoretical Linguistics 2 (QITL-2).

Sangwan, A., & Hansen, J. H. (2009). Leveraging Speech Production Knowledge for Improved Speech Recognition. *Proceedings of the 2009 IEEE Workshop on Automatic Speech Recognition & Understanding.* Merano, Italy: IEEE Press.

Santorini, B. (1990). *Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd Printing).* Philadelphia, PA: University of Pennsylvania.

Sapir, E. (1921). *Language: An Introduction to the Study of Speech.* New York, NY: Harcourt, Brace and Company.

Sarikaya, R., Afify, M., Deng, Y., Hakan, E., & Gao, Y. (2008). Joint Morphological-Lexical Language Modeling for Processing Morphologically Rich Languages with Application to Dialectal Arabic. *IEEE Transactions on Audio, Speech, and Language Processing, 16*(7), 1330-1339.

Saul, L. K., & Jordan, M. I. (1999). Mixed Memory Markov Models: Decomposing Complex Stochastic Processes As Mixtures Of Simpler Ones. *Machine Learning, 37*(1), 75-87.

Savova, V., & Peshkin, L. (2007). Dependency Parsing with Dynamic Bayesian Network. *Proceedings of the National Conference on Artificial Intelligence Vol. 20 No. 3.* Menlo Park, CA: AAAI Press.

Scally, J. R., Cassimatis, N. L., & Uchida, H. (2011). Worlds as a Unifying Element of Knowledge Representation. *Advances in Cognitive Systems: Papers from the 2011 AAAI Fall Symposium (FS-11-01).* Arlington, VA: AAAI Press.

Schaffner, C. (2004). Metaphor and translation: some implications of a cognitive approach. *Journal of Pragmatics, 36*(7), 1253-1269.

Schank, R. C. (1972). Conceptual dependence: A theory of natural language understanding. *Cognitive Psychology, 3*(4), 552-631.

Schank, R. C., & Abelson, R. P. (1975). *Scripts, Plans, and Knowledge.* New Haven, CT: Yale University.

Scheffer, T., Decomain, C., & Wrobel, S. (2001). Active Hidden Markov Models for Information Extraction. *Lecture Notes in Computer Science: Advances in Intelligent Data Analysis, 2189*, 309-318.

Scheutz, M., Cantrell, R., & Schermerhorn, P. (2011). *Toward Human-Like Task-based Dialogue Processing for HRI.* Somerville, MA: Tufts University.

Schlangen, D., Baumann, T., & Atterer, M. (2009). Incremental Reference Resolution - The Task, Metrics for Evaluation, and a Bayesian Filtering Model that is Sensitive to Disfluencies. *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue.* London, UK: Association for Computational Linguistics.

Schmid, H. (2000). *Unsupervised Learning of Period Disambiguation for Tokenization.* Stuttgart, Germany: Internal Report, IMS, University of Stuttgart.

Scholkopf, B., Smola, A., & Muller, K.-B. (1997). Kernel principal component analysis. *Proceedings of Artificial Neural Networks--ICANN'97* (pp. 583-588). Berlin, Germany: Springer Berlin Heidelberg.

Schuemie, M. J., & Mons, B. (2005). Word Sense Disambiguation in the Biomedical Domain - An Overview. *Journal of Computational Biology, 12*(5), 554-565.

Schuler, W., Wu, S., & Schwartz, L. (2009). A framework for last incremental interpretation during speech decoding. *Computational Linguistics, 35*(3), 313-343.

Schwitter, R., & Tilbrook, M. (2004). Controlled Natural Language meets the Semantic Web. *Proceedings of the Australasian Language Technology Workshop.* Sydney, Australia: Australasian Language Technology Association.

Scoble, R. (2010, April 28). *Breaking News: Siri bought by Apple.* Retrieved December 13, 2012, from http://scobleizer.com/2010/04/28/breaking-news-siri-bought-by-apple/

Scudder, H. (1965). Probability of Error of Some Adaptive Pattern-Recognition Machines. *IEEE Transactions on Information Theory, 11*(3), 363-371.

Searle, J. (1980). Minds, Brains, and Programs. *Behavioral and Brain Science, 3*(3), 417-457.

Searle, J. (2011, February 23). Watson Doesn't Know It Won on 'Jeopardy!'. *The Wall Street Journal*. Retrieved March 2, 2013 from: http://online.wsj.com/news/articles/SB10001424052748703407304576154313126987674

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys (CSUR), 34*(1), 1-47.

Selman, B., Kautz, H., & Cohen, B. (1993). Local Search Strategies for Satisfiability Testing. *Cliques, coloring, and satisfiability: Second DIMACS implementation challenge* (pp. 521-532). Piscataway, NJ: DIMACS.

Sen, P., Maglaris, B., Kikli, N.-E., & Anastassiou, D. (1989). Models for packet switching of variable-bit-rate video sources. *IEEE Journal on Selected Areas in Communication, 7*(5), 865-869.

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications.* Geneva, Switzerland: Association for Computational Linguistics.

Seymore, K., McCallum, A., & Rosenfeld, R. (1999). Learning Hidden Markov Model Structure for Information Extraction. *AAAI-99 Workshop on Machine Learning for Information Extraction.* Orlando, FL: AAAI Press.

Sha, F., & Pereira, F. (2003). Shallow parsing with conditional random fields. *Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (Volume 1).* Baltimore, MD: Association for Computational Linguistics.

Shalizi, C. R. (2011, July 11). *Maximum Entropy Models (MaxEnt).* Retrieved February 13, 2013, from http://vserver1.cscs.lsa.umich.edu/~crshalizi/notebooks/max-ent.html

Shanahan, M. (1997). *Solving the frame problem: a mathematical investigation of the common sense law of inertia.* Cambridge, MA: MIT Press.

Shanahan, M., & Witkowski, M. (2004). Event calculus planning through satisfiability. *Journal of Logic and Computation, 14*(5), 731-745.

Shannon, C. E. (1951). Prediction and Entropy of Printed English. *Bell Systems Technical Journal, 30*(1), 50-64.

Shannon, C. E. (2001). A Mathematical Theory of Communication. *ACM SIGMOBILE Mobile Computing and Communciation Review, 5*(1), 3-55.

Shapiro, S. C., & Johnson, F. L. (2000). Automatic Belief Revision in SNePS. In C. Baral, & M. Truszczynski (Ed.), *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning (NMR2000).* Breckenridge, CO: Principles of Knowledge Representation and Reasoning.

Shapiro, S. C., & Neal, J. G. (1982). A Knowledge Engineering Approach to Natural Language Understanding. *Proceedings of the 20th Annual Meeting on Association for Computational Linguistics.* Morristown, NJ: Association for Computational Linguistics.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis.* Cambridge, UK: Cambridge University Press.

Shen, E. (2010, June). *Comparison of online machine translation tools.* Retrieved May 30, 2013, from tcworld: http://www.tcworld.info/e-magazine/translation-and-localization/article/comparison-of-online-machine-translation-tools/

Shen, H., & Sarkar, A. (2005). Voting between multiple data representations for text chunking. *Lecture Notes in Computer Science: Advances in Artificial Intelligence, 3501*, 389-400.

Shieber, S. M. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy, 8*(3), 333-343.

Sholes, L. (1878). United States Patent No. 207559, Washington, DC: United States Patent and Trademark Office.

Silverman, M. (2011, February 17). *Why Watson's win doesn't make humanity obsolete—yet*. (CNN) Retrieved March 10, 2013, from http://www.cnn.com/2011/TECH/innovation/02/17/watson.silverman/index.html

Simon, H. A. (1956). Rational choice and the structure of the environment. *Psychological Review, 63*(2), 129-138.

Simon, H. A. (1995). Artificial Intelligence: An Empirical Science. *Artificial Intelligence, 77*(1), 95-127.

Simon, H. A. (1996). *The Sciences of the Artificial.* Cambridge, MA: MIT Press.

Simon, H. A., & Kaplan, C. (1989). *Foundations of Cognitive Science.* Cambridge, MA: MIT Press.

Singhal, A. (2012, May 16). *Introducing the Knowledge Graph: Things, Not Strings*. Retrieved July 14, 2013, from Official Blog of Google: http://googleblog.blogspot.co.uk/2012/05/introducing-knowledge-graph-things-not.html

Singla, P., & Domingos, P. (2006). Entity Resolution with Markov Logic. *Proceedings of the 6th International Conference on Data Mining (ICDM'06).* Hong Kong, China: IEEE Press.

Singla, P., & Domingos, P. (2006). Memory-efficient inference in relational domains. *Proceedings of the Twenty-First National Conference on Artificial Intelligence.* Boston, MA: AAAI Press.

Skinner, B. F. (1938). *The Behavior of Organisms: An Experimental Analysis.* Cambridge, MA: B.F. Skinner Foundation.

Skoyles, J. R. (2000). Gesture, Language Origins, and Right Handedness. *Psychology, 11*(24).

Skut, W., & Brants, T. (1998). A Maximum-Entropy Partial Parser for Unrestricted Text. *Proceedings of the 6th Workshop on Very Large Corpora.* Montreal, Canada: LREC.

SkyPhrase. (2013, September 29). *SkyPhrase - Data with Natural Language*. Retrieved October 1, 2013, from https://skyphrase.com/

SkyPhrase. (2013). *Skyphrase: Making data simple with natural language*. Retrieved March 11, 2013, from SkyPhrase Blog: http://blog.skyphrase.com/

Sleator, D. D., & Temperley, D. (1993). Parsing English with a Link Grammar. *Proceedings of the Third International Workshop on Parsing Technologies (IWPT 93).* Tilburg, Netherlands: Association for Computational Linguistics.

Slobodchikoff, C. N., Perla, B. S., & Verdolin, J. L. (2009). *Prairie Dogs: Communication and Community in an Animal Society.* Cambridge, MA: Harvard University Press.

Smith, A., & Veneris, A. (2005). Fault diagnosis and logic debugging using boolean satisfiability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 24*(10), 1606-1621.

Smith, R. E., & Swinyard, W. R. (1988). Cognitive Response to Advertising and Trial: Belief Strength, Belief Confidence, and Product Curiosity. *Journal of Advertising, 17*(3), 3-14.

SnePS Research Group. (2011, June 14). *SNePS Research Group Home Page*. (SUNY Buffalo) Retrieved January 24, 2013, from http://www.cse.buffalo.edu/sneps/

Socher, R., Lin, C. C., Ng, A. Y., & Manning, C. D. (2011). Parsing Natural Scenes and Natural Language with Recursive Neural Networks. *Proceedings of the 26th International Conference on Machine Learning (ICML). Vol. 2. No. 8.* Bellevue, WA: International Machine Learning Society.

Soderland, S., & Lehnert, W. (1994). Corpus-driven knowledge acquisition for discourse analysis. *Proceedings of the 12th National Conference on Artificial Intelligence (Volume 1).* Seattle, WA: AAAI Press.

Solso, R. L., MacLin, K., & MacLin, O. H. (1995). *Cognitive psychology.* Boston, MA: Allyn and Bacon.

Sporleder, C., & Lapata, M. (2005). Discourse Chunking and its Application to Sentence Compression. *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing.* Vancouver, Canada: Association for Computational Linguistics.

SRI. (2010). *PAL: Personalized Assistant that Learns (Overview)*. Retrieved March 10, 2013, from https://pal.sri.com/Plone/framework

Stabler, E. P. (2004). Varieties of crossing dependencies: Structure dependence and mild context sensitivity. *Cognitive Science, 28*(5), 699-720.

Stanford University NLP Group. (2012, July 10). *Stanford Parser*. Retrieved March 10, 2013, from http://nlp.stanford.edu:8080/parser/

Starr, B. (2013, June 20). *Search, Answers & Knowledge Graphs Galore*. Retrieved August 1, 2013, from Search Engine Land: http://searchengineland.com/search-answers-knowledge-graphs-galore-163217

Statos, K. (2012). *Unsupervised Learning 101: the EM for the HMM.* New York, NY: Columbia University.

Steedman, M. (2010, October 27). *Some Important Problems in Natural Language Processing [PDF Document]*. Retrieved June 11, 2013, from The Hamming Seminars: http://www.inf.ed.ac.uk/research/programmes/hamming/Steedman.pdf

Steinwart, I., & Christmann, A. (2008). *Support Vector Machines.* Berlin, Germany: Springer.

Stevenson, M., & Wilks, Y. (2003). Chapter 13: Word-sense Disambiguation. In *The Oxford Handbook of Computational Linguistics* (pp. 249-265). Oxford, UK: Oxford University Press.

Stolcke, A. (1995). An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities. *Computational Linguistics, 21*(2), 165-201.

Stone, G. O., Vanhoy, M., & Van Orden, G. C. (1997). Perception Is a Two-Way Street: Feedforward and Feedback Phonology in Visual Word Recognition. *Journal of Memory and Language, 36*(3), 337-359.

Stoyanov, V., Cardie, C., Gilbert, N., Riloff, E., Buttler, D., & Hysom, D. (2010). *Reconcile: A Coreference Resolution Research Platform.* Ithaca, NY: Cornell University.

Strickland, D. (1997). Virtual Reality for the Treatment of Autism. *Studies in Health Technology and Informatics, 44*, 81-86.

Suddendorf, T., & Corballis, M. C. (1997). Mental time travel and the evolution of the human mind. *Genetic, social, and general psychology monographs, 123*(2), 133-167.

Sullivan, D. (2006, October 5). *Hello Natural Language Search, My Old Over-Hyped Search Friend*. (Search Engine Watch) Retrieved March 10, 2013, from http://searchenginewatch.com/article/2057632/Hello-Natural-Language-Search-My-Old-Over-Hyped-Search-Friend

Sun, R. (1997). Learning, action, and consciousness: A hybrid approach towards modeling consciousness. *Neural Networks: Special Issue on Consciousness, 10*(7), 1217-1331.

Sun, R. (2003). *A Tutorial on CLARION 5.0.* Retrieved December 13, 2013, from http://www.cogsci.rpi.edu/public_html/rsun/sun.tutorial.pdf

Sun, R., Appoloni, B., & Roy, A. (2002). *The Panel on Principles of New Connectionism.* Honolulu, HI: International Joint Conference on Neural Networks (IJCNN'02).

Sun, X., Morency, L.-P., Okanohara, D., & Tsujii, J. (2008). Modeling Latent-Dynamic in Shallow Parsing: A Latent Conditional Model with Improved Inference. *Proceedings of the 22nd International Conference on Computational Linguistics (Volume 1).* Mumbai, India: Assocation for Computational Linguistics.

Sundar, N. (2013). *First Steps in Going from DeepQA to DynamicQA.* Troy, NY: Rensselar Polytechnic Institute.

Sundar, N., Bringsjord, S., & Licato, J. (2013). On Deep Computational Formalization of Natural Language. *Proceedings of AGI-13: Formal Magic.* Osnabrueck, Germany: AGI-13.

Tabor, W. (2009). A dynamical systems perspective on the relationship between symbolic and non-symbolic computation. *Cognitive neurodynamics, 3*(4), 415-427.

Tabor, W., Galantucci, B., & Richardson, D. (2004). Effects of Merely Local Syntactic Coherence on Sentence Processing. *Journal of Memory and Language, 50*(4), 355-370.

Taleb, N. (2005). *Fooled by Randomness: The hidden role of chance in life and in the markets.* New York, NY: Random House Trade Paperbacks.

Talmy, L. (1988). Force Dynamics in Language and Cognition. *Cognitive Science, 12*(1), 49-100.

Tang, L. P. (2003). *Integrating Top-down and Bottom-up Approaches in Inductive Logic Programming: Applications in Natural Language Processing and Relational Data Mining.* Austin, TX: University of Texas.

Tapanainen, P., & Voutilainen, A. (1994). Tagging accurately: Don't guess if you know. *Proceedings of the 4th Conference on Applied Natural Language Processing.* Stuttgart, Germany: Association for Computational Linguistics.

Taskar, B., Guestrin, C., & Koller, D. (2003). Max-Margin Markov Networks. *Proceedings of the Neural Information Processing Systems Conference (NIPS03).* Vancouver, Canada: Neural Information Processing Systems Foundation.

Thagard, P. (2012). *Cognitive Science*. (Stanford Encyclopedia of Philosophy) Retrieved December 13, 2013, from http://plato.stanford.edu/entries/cognitive-science/

Thompson, S. (2010). *The Intimate Ape: Orangutans and the Secret Life of a Vanishing Species.* New York, NY: Citadel.

Tobler, P. N. (2006). Human Neural Learning Depends on Reward Prediction Errors in the Blocking Paradigm. *Journal of Neurophysiology, 95*(1), 301-310.

Toutanova, K., & Manning, C. D. (2000). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.* Hong Kong: Association for Computational Linguistics.

Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (Volume 1).* Edmonton, Canada: Association for Computational Linguistics.

Tribble, A., & Rose, C. (2006). Usable Browsers for Ontological Knowledge Acquisition. *Proceedings of CHI'06: Extended Abstracts on Human Factors in Computing Systems.* Montreal, Canada: ACM.

Truyen, T. T., Phung, D. Q., Bui, H. H., & Venkatesh, S. (2009). Hierarchical Semi-Markov Conditional Random Fields for Recursive Sequential Data. *Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems.* Vancouver, Canada: Neural Information Processing Systems Foundation.

Tugui, A. (2004). Reflections on the Limits of Artificial Intelligence. *Ubiquity, 5*(38), 2-2.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society, 42*(2), 230-265.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind, 59*(236), 433-460.

Turney, P. D. (2006). Similarity of semantic relations. *Computational Linguistics, 32*(3), 379-416.

Turney, P. D., & Littman, M. L. (2005). Corpus-based learning of analogies and semantic relations. *Machine Learning, 60*(1-3), 251-278.

Tversky, A., & Kahneman, D. (1973). Availability: A heuristic for judging frequency and probability. *Cognitive Psychology, 5*(2), 207-232.

Tyre, P. (2012, September 19). *The Writing Revolution.* Retrieved June 2, 2013, from The Atlantic: http://www.theatlantic.com/magazine/archive/2012/10/the-writing-revolution/309090/

Uchida, H. (2013, October 4). Human-Level Natural Language Understanding: False Progress and Real Challenges. (P. G. Bignoli, Interviewer)

Ultsch, A., & Siemon, H. P. (1990). Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. *Proceedings of the International Neural Network Conference (INNC'09)* (pp. 305-308). Paris, France: IEEE Press.

Vadas, D., & Curran, J. (2007). Adding Noun Phrase Structure to the Penn Treebank. *Association for Computational Linguistics Annual Meeting, 45*(1), 753-809.

Van Aken, J. R. (2011, May 31). *A Statistical Algorithm for Word Segmentation.* Retrieved August 1, 2013, from arXiv: http://arxiv.org/ftp/arxiv/papers/1105/1105.6162.pdf

Veronis, J., & Langlais, P. (2000). Evaluation of parallel text alignment systems: ARCADE. In *Parallel Text Processing* (pp. 369-388). Berlin, Germany: Kluwer Academic Publishers.

Vishwanathan, S. V., Schraudolph, N. N., Schmidt, M. W., & Murphy, K. P. (2006). Accelerated Training Conditional Random Fields with Stochastic Gradient Methods. *Proceedings of the 23rd International Conference on Machine Learning* (pp. 969-976). Pittsburgh, PA: ACM.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory, 13*(2), 260-269.

Vlachos, A. (2011). Evaluating unsupervised learning for natural language processing tasks. *Proceedings of the First Workshop on Unsupervised Learning in NLP.* Edinburgh, Scotland: Association for Computational Linguistics.

Voorhees, E. M. (1999). The TREC-8 question answering track report. *Proceedings of TREC (Volume 8).* Gaithersburg, MD: National Institute of Standards and Technology.

Voorhees, E. M., & Harman, D. (1998). The text retrieval conferences (TRECs). *Proceedings of a Workshop held at Baltimore, Maryland on October 13-15.* Baltimore, MD: Association for Computational Linguistics.

Wagenmakers, E.-J., Farrell, S., & Ratcliff, R. (2005). Human Cognition and a Pile of Sand: A Discussion on Serial Correlations and Self-Organized Criticality. *General Journal of Experimental Psychology, 134*(1), 108.

Wagner, J. (2012). *Detecting Grammatical Errors with Treebank-Induced Probabilistic Parsers*. Dublin, Ireland: Dublin City University.

Walder, C. J., Kootsookos, P. J., & Lovell, B. C. (2003). Towards a Maximum Entropy Method for Estimating HMM Parameters. *Proceedings of the 2003 APRS Workshop on Digital Image Computing. 1.* Brisbane, Australia: Australian Pattern Recognition Society.

Wang, J., & Brown, M. S. (2007). Automated Essay Scoring Versus Human Scoring: A Comparative Study. *The Journal of Technology, Learning, and Assessment, 6*(2), 1-29.

Wang, K., Thrasher, C., Viegas, E., Li, X., & Hsu, P. (2010). An Overview of Microsoft Web N-gram Corpus and Applications. *Proceedings of the NAACL HLT 2010 Demonstration Session.* Los Angeles, CA: Association for Computational Linguistics.

Wang, R., & Zhang, Y. (2010). Hybrid Constituent and Dependency Parsing with Tsinghua Chinese Treebank. *Proceedings of the seventh international conference on Language Resources and Evaluation.* Valletta, Malta: LREC.

Watanabe, S. (1985). *Pattern recognition: human and mechanical.* Hoboken, NJ: John Wiley & Sons, Inc.

Weber, R. (2011). *Markov Chains - Course Notes.* Retrieved March 25, 2013, from http://www.statslab.cam.ac.uk/~rrw1/markov/M.pdf

Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemcial Information and Computer Science, 28*(1), 31-36.

Weizenbaum, J. (1966). ELIZA--a computer program for the study of the natural language communication between man and machine. *Communications of the ACM, 9*(1), 36-45.

Welch, L. R. (2003). Hidden Markov Models and the Baum-Welch Algorithm. *IEEE Information Theory Society Newsletter, 53*(4), 1-14.

Whorf, B. L. (1942). Language Mind and Reality. *The Theosophist, 63*(1), 281-291.

Wiegerinck, W. (2005). Modeling Bayesian Networks by Learning from Experts. *Proceedings of BNAIC 2005.* Brussels, Belgium: Koninklijke Vlaamse Academie van Belie voor Wetenschappen en Kunsten.

Wikipedia. (2013, February 7). *John Smith (name)*. (Wikipedia) Retrieved March 10, 2013, from http://en.wikipedia.org/wiki/John_Smith_(name)

Wikipedia. (2013, February 28). *Natural Language Processing*. (Wikipedia) Retrieved March 4, 2013, from http://en.wikipedia.org/wiki/Natural_language_processing

Wikipedia. (2013, February 28). *Powerset (company).* Retrieved March 10, 2013, from http://en.wikipedia.org/wiki/Powerset_(company)

Wikpedia. (2013, March 6). *Siri (Software)*. Retrieved March 10, 2013, from http://en.wikipedia.org/wiki/Siri_(software)

Wilks, Y., & Brewster, C. (2009). Natural Language Processing as a Foundation of the Semantic Web. *Foundations and Trends in Web Science, 1*(4), 199-327.

Williams, J. (2012). Extracting fine-grained durations for verbs from Twitter. *Proceedings of ACL 2012 Student Research Workshop.* Jeju Island, Korea: Association for Computational Linguistics.

Williams, J., & Katz, G. (2012). Extracting and modeling durations for habits and events from Twitter. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers (Voume 2).* Jeju Island, Korea: Association for Computational Linguistics.

Winograd, T. (1971). *Procedures as a Representation for Data in a Computer Program for Understanding Natural Language.* Cambridge, MA: MIT Artificial Intelligence Lab.

Winston, P. H. (1970). *Learning Structural Descriptions from Examples*. (Doctoral Dissertation). Massachusetts Institute of Technology, Cambridge, MA.

Wolfram Research. (2013). *Markov Chains*. Retrieved September 15, 2013, from Wolfram Demonstrations Project: http://demonstrations.wolfram.com/search.html?query=Markov Chains

Wolfram Research. (2013, September 29). *Wolfram|Alpha: Computational Knowledge Engine*. Retrieved October 2, 2013, from http://www.wolframalpha.com/

Wolfram. (n.d.). *Wolfram Mathematica: Technical Computing Software*. Retrieved March 10, 2013, from http://www.wolfram.com/mathematica/

WolframAlpha. (n.d.). *Wolfram|Alpha Frequently Asked Questions*. Retrieved March 10, 2013, from http://www.wolframalpha.com/faqs.html

World Wide Web Consortium. (2013, October 29). *RDF Current Status.* Retrieved November 10, 2013, from W3C: http://www.w3.org/standards/techs/rdf#w3c_all

Xiao, J., Zou, L., & Li, C. (2007). Optimization of Hidden Markov Model by a Genetic Algorithm for Web Information Extraction. *Proceedings of the 2007 International Conference on Intelligent Systems and Knowledge Engineering.* Chengdu, China: Springer.

Yarowsky, D. (2000). Hierarchical decision lists for word sense disambiguation. *Computers and the Humanities, 34*(1), 179-186.

Yellen, D. (2009, October 05). *See Spot Read: Willow the Dog Understands Written Commands.* Retrieved December 13, 2012, from http://www.peoplepets.com/people/pets/article/0,,20493186,00.html

Yelp. (2013, September 29). *Yelp.com*. Retrieved October 1, 2013, from http://www.yelp.com/

Yli-Jyrä, A. (2005). Approximating Dependency Grammars Through Intersection of Regular Languages. In *Implementation and Application of Automata* (pp. 281-292). Kingston, Canada: 9th International Conference, CIAA 2004.

Yu, S.-Z. (2010). Hidden semi-Markov models. *Artificial Intelligence, 174*(2), 215-243.

Zelenko, D., Aone, C., & Richardella, A. (2003). Kernel methods for relation extraction. *The Journal of Machine Learning Research, 3*, 1083-1106.

Zhang, C., Qui, J., Zheng, S., & Yang, X. (2012). Cross-Layered Hidden Markov Modeling for Surveillance Event Recognition. *Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW).* Melbourne, Australia: IEEE Press.

Zhang, H., & Gu, C. (2011). *Support Vector Machines versus Boosting.* Berkeley, CA: Department of EECS, University of California.

Zhou, D., Zhang, M., Ji, D. H., & Zhu, Q. M. (2007). Tree Kernel-based Relation Extraction with Context-Sensitive Structured Parse Tree Information. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.* Prague, Czech Republic: Association for Computational Linguistics.

Zhou, Y., Weng, F., Wu, L., & Schmidt, H. (2003). A Fast Algorithm for Feature Selection in Conditional Maximum Entropy Modeling. *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing.* Sapporo, Japan: Association for Computational Linguistics.

Zhou, Z.-H., Wu, J., & Tang, W. (2002). Ensemble Learning. *Artificial Intelligence, 137*(1), 239-263.

Zipf, G. (1950). *Human Behavior and the Principle of Least Effort.* Boston, MA: Addison Wesley.

Zushi, M. (2001). *Long-Distance Dependencies.* London, UK: Routledge.

24. Appendix 1: Summary of Knowledge-Lean Deficiencies and Possible Solutions

Recall that the intent of this document is to assess the various approaches to human-level NLP (and, by proxy, human-level AI) that are currently being pursued by researchers in the field. Almost the entirety of the NLP research community is invested in knowledge-lean, machine-learning-based models of language. Because it has been shown that traditional knowledge-engineered systems are not likely to be extendable to human-level AI for a variety of reasons, it is now necessary to determine whether knowledge-lean platforms are an adequate alternative. Having examined the statistical evaluation of knowledge-lean, machine-learning approaches to NLP in some detail, it is necessary to determine how to evaluate this portion of the field. On one hand, many of the general principles of knowledge-lean systems seem in line with the characteristics of human cognition: both are flexible, robust, efficient, and scalable. However, considering, at least, the way that such models are evaluated, it appears that knowledge-lean approaches are also incompatible with human-level performance.

Human cognition is deeply involved with context and complex, global structures, whereas machine-learn approaches are evaluated on how well they model patterns in data that are simple, local, and as context-independent as possible. Furthermore, one point that deserves to be considered is that the metrics analyzed above are mean to gauge performance on what are arguably highly artificial and controlled tasks. By analogy, the fact that humans can solve problems under unrealistic, artificial laboratory conditions does not mean that they are incapable of reasoning in a more organic way in more natural situations. The metrics discussed above are not intended to measure holistic language competence, at the human-level or not. Many of the individual aspects of language are governed by local structures, such as phrases. All that is clear is that the types of NLP tasks above and the metrics that have been designed to evaluate them are not able to (and probably were never intended to) scale up to human-level performance. Almost all of the literature in the NLP field that is accompanied by actual implementations focuses on these limited types of systems. It is possible that the knowledge-lean approaches are capable of more sophisticated endeavors. Thus, to determine whether or not knowledge-lean systems are fit for human-level NLP projects, it is necessary to investigate the

larger-scale, more integrated approaches that attempt to provide human-level performance, even if they might not be evaluable by formal metrics.

24.1 Hidden Markov Models and Limited Memory Modeling

Even a cursory overview of probabilistic modeling formalisms would surely encounter at least some paper about Hidden Markov Models (HMMs) (Rabiner, 1989). HMMs were developed by the Russian Mathematician Andrey Markov in the early twentieth century (Markov, 1971) as a way to model certain kinds of stochastic processes. Stochastic processes act as generators for observable data such that some information about the operation of the process is not available, which leads to some aspect of its operation appearing random.

As a simple example, the tossing of a single coin can be thought of as a stochastic process, since, for all intents and purposes, the behavior of the coin is unpredictable from its initial conditions. While it may be possible to construct detailed physical models of the coin, there is enough uncertainty in this system that a casual participant in a coin-prediction experiment would be expected to perform at the chance level. It is possible to make a model of a stochastic process by examining patterns in the observed behavior (i.e. data) of the system (i.e. coin) and then relating these patterns to the structural components of the data generator. Consider the same example as before. The generator in this case can be considered to be a single random variable that corresponds to the coin. This random variable either produces a head or a tail datum for each coin flip. To create a model of this system, there is only one parameter that needs to be set, namely, the probability of producing a head datum during each coin toss. By examining a large number of coin tosses, it is possible to produce an estimate of this parameter, as it would be expected that, over large intervals of time, the observable patterns in the data would correspond to some fixed parameter of the system.

For a stochastic process where the structure of the data generator (e.g. the single coin in the example above) is completely known, it is possible to create an observable model for the system, where the states of the generator are explicitly constructed. However, most complex systems, including those that generate natural language data, are far too complex to be able to reconstruct the states of the system precisely. In such cases,

a “hidden” model (e.g. a HMM) needs to be created, in which the observable states of model behavior are supplemented by a number of indefinite structural states. By learning different relationships between these hidden states and between hidden states and observed states, it is possible to create an approximation of the generator of the observed data. Conceptually, hidden models operate by supposing that the generator itself is totally composed of the hidden states of the model. The operation of the generator is conceived of as different sequences of passing through these hidden states. For each datum that is generated, the generator is in one of the hidden states—the state that is said to emit that datum.

In the basic HMM idiom, a stochastic process (i.e. the generator of some random data) is considered to consist of a finite set of states. Some general information about the operation of HMMs follows. HMMs are a type of graphical model (Lauritzen, 1996), which means that they can be considered naturally through the metaphor of a set of nodes connected with a set of links. In the case of an HMM, the set of nodes is broken into two subsets: a set of nodes corresponding to observed states and a set of node corresponding to hidden states. There are three parameters that need to established in order to produce an HMM for a stochastic process. First, each of the hidden states in an HMM transitions to other hidden states according to some probability. These values are cumulatively called the transition probabilities. Second, each of the hidden states has emission probabilities that determines the probability of producing each of the observable states if the system is currently in that particular hidden state. These values are cumulatively called the emission probabilities. A third parameter, called the initial probabilities, is the set of values that define the probability that the system will start in a particular hidden state. In order make tractable the actual learning and inference processes of computational models, it is necessary for every modeling formalism to make some kind of simplifying assumptions. Markov models make use of the Markov assumption, which states that the next transition between states is only contingent on the current state. This assumption greatly reduces the computational resources required to create and utilize Markov models.

The potential for HMMs to study the properties of natural languages was clear even to Markov, who imagined using them to study the distribution patterns of

consonants and vowels in Russian literature (Markov, 1913). Historically, HMMs have been among the most popular methods for statistical NLP in particular. Some of the first computerized HMMs were used to develop applications for automated speech recognition systems (Malode & Sahare, 1963). Although more modern methods have become more in vogue in recent times, especially models based on Conditional Random Fields (Lafferty, McCallum, & Pereira, 2001) or Markov Random Fields (Kindermann & Snell, 1980), HMMs are still used for a large number of NLP tasks. Some examples of these applications are: Part of Speech Tagging (Brants, 2000), Shallow Parsing (Shen & Sarkar, 2005), and Information Retrieval (Seymore, McCallum, & Rosenfeld, 1999). Because they are so central to the statistical NLP field, it is worth looking into how HMMs are capable of modeling these kinds of stochastic processes.

There are also a large number of extensions to basic HMMs, such as Factorized HMMs (Ghahramani & Jordan, 1997) and Hierarchical HMMs (Fine, Singer, & Tishby, 1998). Given the amount of information about HMMs, it is clear that some preliminary work must be done to analyze these different models and to categorize their shortcomings and highpoints with respect to solving NLP problems. To complicate matters further, it turns out that the final product of training a HMM results in a model of equivalent computational power to that of a Non-Deterministic Finite State Transducer. A Finite State Transducer is an extension to a Finite State Machine (Hopcroft, Motwani, & Ullman, 1979), in which there is an output alphabet in addition to the standard FSM input alphabet and each of the state transitions of the transducer produces some output in the output alphabet. Thus, it is also necessary to determine how HMMs relate to other kinds of statistical and non-statistical methods for solving NLP problems.

24.2 Maximum Entropy

Even though the general design of HMMs is as simple as it is effective at being applied to a large number of diverse problems and even though this basic design has proven its relative fitness for modeling stochastic processes through the sheer length of the time during which they have been deployed, there are a number of shortcomings to HMMs that have motivated the creation of alternative modeling formalisms. One of the most prominent of these alternatives is the Maximum Entropy (ME) framework

(Ratnaparkhi, 1997). In itself, ME models are not sequential classifiers like HMMs, but instead act as a way to classify data at a particular time slice. It turns out that many NLP problems are naturally solved as classification problems. Part of Speech tagging can be thought of as classifying words into part of speech types (e.g. *dog* could be classified as either a noun or a verb). Alternatively, parsing can be thought of as classifying sentences into abstract parse structures. In general, the task of classification is defined as mapping data elements to the probability of that element's membership in each of a set of predefined categories.

ME models approach the task of classification via the specification of patterns of features in a data set that are presupposed to be relevant to the type of classification being performed. These features specify some kind of environment around the current data element that can be used for the purpose of classification. For example, if the current classification task is to determine the most likely part of speech of the word *dog* in a sentence, the features might involve examining the words and tags that surround that word. In the sentence, *the dog is barking*, the features might include the identities of the words before and after *dog*, f1 = *the* and f2 = *is*. Using such features, the ultimate classification is based on how words with similar contexts have been classified during training. If all words in the training set with this particular environment (i.e. preceded directly by *the* and followed directly by *is*) have been classified as a noun, then this occurrence of *dog* would almost certainly also be classified as a noun. Since each feature only partially describes the environment of a datum, it is possible for multiple features to apply to a given datum. The cumulative set of activated features for a particular datum can therefore contain conflicting classification mappings. It is then necessary to provide some weighting to the features so that the most likely classification is inferred. Efficient methods for estimating the weights of features have been identified, such as Iterative Scaling (Darroch & Ratcliff, 1972).

Clearly, there are a huge number of possible textual environments for words in any given text, but it is possible to increase tractability by using more efficient kinds of features. One way to reduce the number of features would be to examine the part-of-speech tags of the preceding and following words instead of using the exact identities of the words themselves. Under this scheme, the features would be f1 = determiner and f2 =

verb. While this certainly reduces the number of possible environments for words, it also is not as descriptive as an environment that also contained some word-specific environments (e.g. if a word is followed by *are*, then there is a very good chance it is a plural noun). To account for the need to have a rich, but not intractably large, set of features, researchers have developed feature selection algorithms to compute the optimal feature set under a set size threshold (Zhou, Weng, Wu, & Schmidt, 2003). The pool of possible features is defined by feature templates (e.g, all combinations of one word before the current word, all combinations of one word after the current word, etc.). By examining the information gain obtained by the model by adding each feature in isolation, it is possible to estimate the relative utility of each feature. If a minimum gain threshold is established, then weakly informative features will not be considered in the final model.

Like HMMs, Maximum Entropy models play a key role in modern statistical NLP, so it is equally important to delve into some of the strengths and weaknesses of this formalism in greater detail. One of the ways to accomplish this task is to examine the assumptions that are made by this modeling formalism and analyze the impact those assumptions have on the ultimate potential of the model with respect to achieving human-level performance. For instance, the major assumption of the Maximum Entropy framework (c.f. the Markov Assumption described above) is similar in form to Ockham's Razor. In this case, the simplest explanatory model is the one that uses the features to their maximum extent, but considers all other influences on the distribution of data to follow a uniform distribution. At the onset, this seems at odds with the way that human cognition operates, as humans readily use analogical reasoning to solve novel problems in a similar fashion to previously solved ones (Holyoak, Gentner, & Kokinov, 2001).

24.3 Probabilistic Context Free Grammars and Beyond

In order to address some of the shortcomings of the HMM framework, it is necessary to create models that are more expressive than Regular Languages. Indeed, one of the major findings in modern linguistics was Chomsky's proof that natural languages possess the ability to nest phrases recursively and thus are at least as complex as a family of languages known as Context-Free Grammars (Chomsky, 2002). A Context

Free Grammar consists of a set of non-terminal phrase symbols, a set of terminal lexical symbols, and a set of phrase production rules that map a single non-terminal symbol to a string of terminal and non-terminal symbols. For instance, in a language where there all sentences are made up of a noun phrase and a verb phrase, *the dog* is the only noun phrase, and *barks* is the only verb phrase, then the corresponding Context-Free Grammar would be:

$$S \rightarrow NP\ VP$$
$$NP \rightarrow \text{the dog}$$
$$VP \rightarrow \text{barks}$$

However, in natural languages, there are many types of sentence structures and the content of different kinds of phrases essentially can be infinite in variety. This situation motivated the creation of Probabilistic Context Free Grammars (PCFGs) and the associated algorithms required to produce the most likely phrase parse of a lexical sentence (Stolcke, 1995). In a PCFG, each of the production rules is associated with a probability. The most likely parse of a sentence is defined as the one that maximizes the overall probability of the phrase production rules implicated in the generation of the sentence. In an effort to increase the sophistication of context-free parsing, lexicalized grammars were introduced that accommodated the definition of subtypes of phrase symbols based on the lexical head of the phrase (Collins, 1999). The Lexicalized version of the Context-Free Grammar above is:

$$S_{bark} \rightarrow NP\ VP$$
$$NP_{dog} \rightarrow \text{the dog}$$
$$VP_{bark} \rightarrow \text{barks}$$

Despite the fact that PCFGs are far more expressive than standard Markov Models, it turns out that natural languages are thought to be even more complex than the types of languages that can be described by Context-Free Grammars. Studies have shown that at least some natural languages are within the family of context-sensitive languages (Shieber, 1985). Intuitively, this seems sensible, since context plays a large role in the conveyance of meaning in natural languages, beyond even what is possible through lexicalized PCFGs. However, the algorithms required to parse context-free languages in general are P-SPACE complete in their computational complexity (Kuroda,

1964). Since there are no known efficient procedures for solving P-SPACE complete problems, this is a major obstacle for the NLP field. On one hand, it is probably necessary to incorporate context-sensitivity in NLP applications if the human-level of performance is ever to be reached. On the other hand, it is not feasible to create large-scale parsers that are capable of parsing general context-sensitive languages. Researchers have created a compromise in the form of mildly context-sensitive languages (Joshi, Vijay-Shanker, & Weir, 1991), which incorporate some of the features of context-sensitive languages, but are more computationally tractable to analyze. This seems well-motivated because there are context-sensitive languages that contain features that are very much unlike the kinds of features that arise in natural languages (c.f. page 71 of (Rodriguez, 2010)).

However, even context-sensitivity is not enough to characterize every proposed model of human languages. For instance, the Head Phrase Structure Grammar theory is known to describe recursively enumerable languages (Carpenter, 1991), which are tantamount to Turing Machines in complexity. Because it is currently infeasible to implement parsers that work on such general principles, there is occasion to examine how to leverage some of the principles of extra-context-free parsing to advance towards human-level NLP while still avoiding the sharp increase in computational resources to parse the more expressive families of formal languages.

Connectionism and Other Approaches

Beyond the types of statistical NLP formalisms that were described above, there is an additional family of related knowledge-lean approaches that are known as connectionist models (McClelland, 2009). Connectionist models rely on weighting links between nodes to capture the behavior of the generating function of the target data set. Since connectionist models are based (albeit usually loosely) on biological structures, some researchers consider them more adept for achieving human-level performance. However, in practice connectionist approaches tend to be used for different applications than non-connectionist approaches, rather than to supersede them. One of the most common applications for connectionist models is document (Farkas, 1993).

There are many types of connectionist modeling frameworks; perhaps the most widely used of these is the Artificial Neural Network (Rojas, 1996). Neural Networks are

often used to create computational models of actual cognitive development, such as word acquisition (Li, Farkas, & MacWhinney, 2004) or learning to process regular versus irregular verbs (Plunkett & Marchman, 1993). Although, for the purposes of a study on NLP applications, Support Vector Machines (SVMs) (Cortes & Vapnik, 1995)are probably more relevant to discuss in detail. SVMs are a generalization of a simple type of neural network called a Perceptron (Rosenblatt, 1958). Perceptrons are probably most well-known for initially being unable to model non-linearly separable data generator functions, such as logical exclusive OR (XOR) (Minsky & Papert, 1969). However, Multilayer Perceptrons were later developed to handle non-linearity (Hornik, 1991). By examining the operation of Multilayer Perceptrons, it was discovered that data that was non-linear when considered in its original dimensionality could often be projected onto a new dimensional basis that would enable the data to be partitioned by a hyperplane. The SVM formalism provides an efficient way (known as the Kernel Trick) to map data onto these different dimensional projections. Generalizing beyond SVMs, one reaches the models known as Kernel Machines, which include techniques such as Principal Component Analysis (Jolliffe, 2005). Since kernel-based approaches have been increasing in popularity, especially in the field of NLP research, it is necessary to investigate their relative strengths and weaknesses and relate them to the attributes of the other methods described above.

The variety of esoteric formalisms for NLP is so extensive that it extends beyond what can be mentioned in this document. However, Markov Logic (Domingos, Kok, Poon, Richardson, & Singla, 2006) deserves some special consideration because this platform is touted as being the unification of statistical and knowledge-rich AI. Markov Logic networks are capable of representing random fields over a graphical representation of first order logical formulae. This configuration is extremely powerful in terms of expressivity, but it is so complex that only fairly trivial inference routines can be used to find optimal models. Inference in Markov Logic is equivalent to using a Monte Carlo Method (Metropolis & Ulam, 1949) to solve a weighted relational constraint satisfaction problem (Kautz & Selman, 1992). While this sort of procedure is capable of solving a diverse assortment of problems within one single framework, it is not nearly powerful enough to capture all of the nuances of human cognition. Additionally, Markov Logic

requires that all formulae be grounded out (i.e. all variables in expressions are bound to predefined domain objects), which causes a number of difficulties for creating truly adaptable models, such as ones that require the instantiation of new objects during inference. Although Markov Logic has been applied to NLP problems (Poon, 2011), doing so has not proven to make any significant headway towards human-level performance. As will be discussed below, the drawbacks of Markov Logic motivate the use of other techniques to combine knowledge-lean and knowledge-rich processing techniques.

24.4 Cognitive Architectures and the Human Cognitive Substrate

One of the most challenging aspects of human-level NLP research is that both knowledge-lean and knowledge-rich approaches seem to be irreparably deficient. An even greater obstacle is that the Markov Logic formalism, which was designed to leverage the strengths of these two branches of NLP research, has a large number of flaws in its own right. Yet, it seems inevitable that some kind of combination of knowledge-lean and knowledge-rich reasoning will be required to achieve human-level performance. Fortunately, there are other ways to approach human-level NLP that are broad enough to accommodate both knowledge-lean and knowledge-rich techniques. Perhaps the most promising of these possibilities is the application of cognitive architectures (Langley, Laird, & Rogers, 2009) to simulate human-like NLP behavior.

Cognitive Architectures are formalisms for computational modeling of based on principles put forth by Alan Newell (Newell, 1994), who was one of the founders of the field of AI research. There are a large variety of cognitive architectures currently under development, each of which is geared towards modeling human cognition from a specific perspective. Compared with all of the other formalisms described in this paper, Cognitive Architectures are the most adept at combining different kinds of representations and processing algorithms inside of a single unified framework. A large number of diverse Cognitive Architectures have been proposed, such as ACT-R (Anderson J. R., 1996), CLARION (Sun R. , 2003), SOAR (Milnes et al., 1992), and ICARUS (Langley, Laird, & Rogers, 2009). Some of these cognitive architectures have been used to model solutions to NLP problems (Ball, Freiman, Rodgers, & Myers, 2010).

However, due to the fact that the Polyscheme Cognitive Architecture is purposefully designed to accommodate the specification of complex, adaptable algorithms (Cassimatis et al., 2010), it stands out as one of the better candidates for supporting a project aimed at combining the strong points of the various approaches to NLP that have been describing the literature.

Because the human cognitive system is currently the only extant human-level reasoning system, emulating (at least at some level) the processes at work in the human mind is one way to make head roads into developing more sophisticated AI and NLP programs. Therefore, it seems sensible to use the findings of Cognitive Science research to motivate the designs of computational models of all kinds of human behavior, including language processing. If this is true, then a sensible research program would be to perform a meta-analysis of cognitive science research to mine out the broadest trends in the human cognitive system. Given that the ultimate goal of Cognitive Science is to understand how the human mind works in toto and that one of the most successful ways to demonstrate knowledge of a system is to create an artificial replication of it, it seems as though a project combining research in AI with Cognitive Architectures would stand to benefit both fields. In terms of motivation from cognitive research, Polyscheme has a very complete theory about the fundamental components of human thought, which is called the Human Cognitive Substrate.

The upshot of the Human Cognitive Substrate is that all problems that humans can solve are ultimately reducible to combinations of algorithms implemented in a relatively small number of physical reasoning domains (Cassimatis N. L., 2006). Specifically, a proposal for how NLP tasks can be cast as physical reasoning problems has previously been put forth. There are two aspects of research when working with Polyscheme. The first is the implementation of the components of the physical reasoning substrate, such as procedures and representations for reasoning about temporal intervals (Allen, 1983) and the identity of objects (Pasula, Marthi, Milch, Russell, & Shpitser, 2002). Second, once the basic modules have been implemented, it is necessary to create models that make use of those modules to perform inference. Instances of both types of projects are proposed in the sections below.

24.5 Reasoning about Object Equality

Some of the most fundamental aspects of real-world reasoning problems rely on the ability to handle inferences about the equality or non-equality of objects. Studies in human development have shown that tasks, which clearly involve reasoning about equality, such as understanding object permanence (Baillargeon, Spelke, & Wasserman, 1985), are part of the normal repertoire of early cognitive development. There are also a number of philosophical conundrums that involve the equality of objects (Quine, 1950), which implies that humans are at least capable of explicitly thinking about such matters. More abstractly, reasoning about equality is involved in the logical operation of unification (Baader & Snyder, 2001). Analogs to logical unification in human cognition can be found in various forms, such as pattern recognition (Watanabe, 1985). Thus, there is every reason to believe that equality is one of the fundamental human cognitive substrate elements. Since the processing of language is thought to be intimately intertwined with the rest of human cognition (Emmorey, 2001), it follows that equality should factor somehow in language processing tasks.

One of the most obvious roles that reasoning about equality plays in solving NLP problems is its contribution to reference resolution problems (Eckert & Strube, 2000). Many natural languages incorporate pronouns (e.g. *he*, *she*, *they*, etc.) that serve as variables for other entities in the discourse or in the immediate context of the discourse (e.g. *he* could refer to a nearby person not previously mentioned). Mapping pronouns to their intended referents can be considered as a type of variable binding procedure. Clearly, some notion of equality and abject persistence must be in play to support utterances that use pronouns. However, pronouns are not the only kind of reference that requires binding between entities. Every mention of an object throughout a discourse ultimately needs to be mapped to a previous entity in the discourse, to some entity in the immediate environment, or to some entity in the shared cognitive space of the individuals participating in the discourse. Yet, reasoning about identities permeates even deeper into cognitive processing. At the fundamental levels of syntactic parsing, identities between the components of phrase structure rules must be maintained. For instance, given the knowledge that sentences can have the pattern NP VP, it is necessary to consider each recovered NP as a possible binder for the NP of a sentence.

Despite the overwhelming evidence in support of incorporating some facility for reasoning about equality in a cognitive architecture, standard AI approaches typically do not do so. For instance, Markov Logic does not gracefully handle problems that are heavily reliant on reasoning about equality. Other kinds of constraint satisfaction solving routines, such as DPLL (Davis, Logemann, & Loveland, 1962), also have difficulty handling situations in which the underlying equality of objects is in question. This is due to the fact that the equality of entities can lead to the equality of clauses, which is not accounted for by the traditional cost function for models of these theories. The equality of clauses leads to non-monotonic changes in cost during systematic search and violates the assumptions of widely used branch-and-bound heuristics. To address these problems, it is proposed that a formal logical language be defined that incorporates provisions for reasoning about equality. It is also necessary to introduce a new cost function for models expressed in that language. Next, it is proposed that a lazy algorithm based on DPLL be formulated for finding models in that language. Finally, it is proposed that a novel search heuristic based on equality relationships be developed for this algorithm. Experiments will then be conducted to demonstrate how these measures improve performance relative to systems that do not incorporate such features.

24.6 Reasoning about Time

Like equality, reasoning about time is an important aspect of a huge number of problems that humans must solve on a constant basis. General cognitive capabilities, such as mental time travel (Suddendorf & Corballis, 1997), rely directly on the ability to be able to reason about the absolute and relative times at which relevant events occurred. However, there are also many language-processing specific tasks that require temporal reasoning. One of the most obvious applications of temporal reasoning in terms of language understanding is the interpretation of the time structure of event from tenses and adverbs (e.g. yesterday) (Dowty, 1982). Temporal reasoning is also implicated in solving reference resolution problems. Most references in English are anaphora, which means that the initial mention of the entity precedes the pronoun (or other referring expression) that refers to the entity (e.g. *Perrin went to the store. He bought a shirt*). However,

cataphora, in which the referring expression precedes the actual entity, is also possible (e.g. *Because he wanted to be back by 5:00PM, Perrin left in the mid-afternoon.*).

But, similarly to equality reasoning's relationship to language, temporal reasoning has more fundamental roles, as well. In keeping with the physical substrate reduction hypothesis, it is possible to cast linguistic utterances as events. There has been research that details this methodology in great detail (Cassimatis N. L., 2004). However, what is relevant for this proposal is the fact that in order to interpret the significance of events, some understanding of the temporal structure of those events is necessary. For instance, determining causal relationships between events is contingent on knowing which events occurred before other events. To see how this is directly related to NLP tasks, consider the following example. In the sentence *Perrin gave $10 to charity*, the utterance of each word can be considered to be an event. If it is determined that *gave* is the main verb of this sentence and if it the standard English ditransitive verb construction is known, then the most likely parse of this sentence would be to assign *Perrin* as the subject, *$10* as the direct object, and *charity* as the indirect object. This determination is possible in part due to information about the temporal order of the words in the sentence.

Despite the fact that temporal reasoning is extremely important to human cognition, not all approaches to automated reasoning are capable of handling it with grace. Indeed, even though constraint satisfiability (SAT) testing paradigms, such as Markov Logic, have been used effectively in many inference, planning and constraint satisfaction tasks, there are some situations in which this formulation is not sufficient. For instance, since SAT constraints are defined over atomic propositions, domains with state variables that change over time can lead to extremely large search spaces. This poses both memory- and time-efficiency problems for existing SAT algorithms. It is proposed to address these problems by introducing a language that encodes the temporal intervals over which relations occur and an integrated system that satisfies constraints formulated in this language. Temporal intervals are presented as a compressed method of encoding time that results in significantly smaller search spaces. However, intervals cannot be used efficiently without significant modifications to traditional SAT algorithms.

Using the Polyscheme cognitive architecture, it will be possible to create a system that integrates a DPLL-like SAT-solving algorithm with a rule matcher in order to support intervals by allowing new constraints and objects to be lazily instantiated throughout inference. This system will include constraint graphs to compactly store information about temporal and equality relationships between objects. In addition, a memory retrieval subsystem is proposed to guide inference towards minimal models in common sense reasoning problems involving time and change. After these implementations have been completed, it will be possible to perform two sets of evaluations to isolate the contributions of the system's individual components. These tests will demonstrate that both the ability to add new objects during inference and the use of smart memory retrieval result in a significant increase in performance over pure satisfiability algorithms alone and offer solutions to some problems on a larger scale than what was possible before.

24.7 Model Building and Conclusion

Beyond extensions to the Polyscheme framework itself, it is necessary to develop more advanced modeling techniques that can seamlessly incorporate knowledge-rich, semantic information to aid in the processing of natural language utterances. One of the most promising programs for this kind of integration is The Naïve Semantics (NS) framework (c.f. (Dahlgren, 1988)). NS provides some insight into how this process might unfold. Like the Polyscheme Cognitive Architecture, NS holds that mental constructions of word meanings are no different from mental constructions of other forms of 'world knowledge.' This enables semantic and syntactic processing to be combined in a single framework.

One of the key roles of semantic information in NLP is to aid in the disambiguation of syntactic parsing. An early example in NS demonstrates the application of the theory to disambiguate a PP attachment. In the sentence, *John wanted to buy a computer for his child*, the PP *for his child* can attach either to the VP headed by *buy*, thus modifying the buying event, or to the NP headed by *computer*, thus creating a subcategory of computer named *computer for his child*. Since it is unlikely that a speaker would invent an odd categorization during this kind of speech act, a semantic

interpretation of the situation indicates that the PP almost certainly directly attaches to *buy*.

Furthermore, NS holds that natural language text can be propositionalized into statements in FOL. This is compatible with the propositional representation used in the central processing aspects of the Polyscheme Cognitive Architecture. Disambiguation occurs by appealing to naïve theories of different situations (also known as frames) that follow the commonsense reasoning approach to encoding knowledge in AI systems (c.f. (McCarthy, 1987)). To illustrate how the disambiguation occurs, one must examine the contents of a proposed "buy event" frame. In this frame, NS claims that the recipient of a "buy" action must be a sentient. The first issue that this creates is that it is not always straightforward to determine who the recipient of an action is. Consider the following case in which the recipient is elided: *John bought some clothes*.

In the above case, one might reasonably assume (although this is purely contextual) that John bought the clothes for himself. The issue with using basic frames is that they do not contain internal logic, so having default values (or any kind of conditionals) is problematic. However, even a simple default value is not sufficient to infer an elided recipient of a "buy" action. World knowledge about the typical types of situations (i.e. frames) involving the "buying" of different types of objects makes the identity of the recipient in the following case less certain: *John bought some flowers*.

In this case, the NS theory seems to be vindicated in that knowledge about the possible uses of flowers can help in the interpretation of the sentence. However, there are some unresolved issues in using frames to solve such problems. First, there is the large number of frames that are required to capture this information. The most basic implementation would be to have a frame for every distinct type of "buying" event. However, this would almost certainly be wasteful since there are probably related classes of "buying" events (at least on this one isolated dimension--that of the type of object being bought) that could be given a single frame. For instance, there may be two "buying" frames: one where the default recipient is the subject of the "buying" verb and one where the default recipient is some entity other than the subject of the "buying" event. Second, there is a need to be able to more seamlessly incorporate statistical information about the syntactic structure of sentences in order to reduce the number of

combinations between frame contents and basic linguistic constructions. The failure to perform this kind of integration is ultimately what has plagued the knowledge-rich approach to NLP from the inception of the field (Pantel & Pennacchiotti, 2006).

Fortunately, there is an existing project that is similar in some ways to the Naïve Semantics approach, but that has also solved a large number of the issues that have heretofore precluded the large-scale development applications based on such methods. This project is the SkyPhrase API (SkyPhrase, n.d.), which was motivated in large part by the design principles of Polyscheme. Since work done for implementing cognitive substrate elements in Polyscheme can in theory be readily applied to the creation of SkyPhrase models, it is consistent for some work to be done in Polyscheme itself (i.e. the above proposals for temporal and identity reasoning) and other work to be based around SkyPhrase. Through the creation of SkyPhrase models that surpass the performance of leading NLP applications, it is possible to demonstrate the value of using the Cognitive Architecture approach to integrating knowledge-lean and knowledge-rich NLP methods.

25. Appendix 2: Restaurant Domain Corpus

A restaurant in Troy

The restaurant in Troy

The restaurants in Troy

Restaurant in Troy

Restaurants in Troy

Restaurant

Restaurants

A restaurant

The restaurant

The restaurants

This restaurant in Troy

These restaurants in Troy

This restaurant

These restaurants

Those restaurants

My restaurant

Your restaurant

His restaurant

Her restaurant

Her restaurant in Troy

His restaurant in Troy

My restaurant in Troy

Your restaurant in Troy

My restaurants

Your restaurants

His restaurants

Her restaurants

Her restaurants in Troy

His restaurants in Troy

My restaurants in Troy

Your restaurants in Troy

Find restaurants in Troy

Find a restaurant in Troy

Find the restaurants in Troy

Find me a restaurant in Troy

Please find me a restaurant in Troy

Could you find me a restaurant in Troy?

Could please you find me a restaurant in Troy?

Could you please find me a restaurant in Troy?

Could you find me a restaurant in Troy please?

Find for me a restaurant in Troy

Restaurants in Troy, NY

Restaurants in Troy, New York

Restaurants in Troy, New York 12180

Restaurants in Troy, NY, USA

Search for restaurants in Troy, NY

Get restaurants in Troy, NY

Search for restaurants in Troy, NY for me

Give me restaurants in Troy?

Show me restaurants in Troy NY?

Retrieve restaurants in Troy NY?

Look for restaurants in Troy, NY?

Now look for restaurants in Troy, NY?

Are there restaurants in Troy, NY?

How many restaurants in Troy, NY?

How many restaurants are in Troy, NY?

Where are the restaurants in Troy, NY?

Which restaurants are in Troy, NY?

Restaurants in the New York Area

Restaurants in the New York Metro Area

Restaurants in Troy or Cohoes

Restaurants in Troy, NY and Cohoes, NY

Restaurants not in Troy

Restaurants not in Troy or Cohoes

Restaurants not in Troy and Cohoes

No restaurants in Troy, NY

Restaurants except in Troy

Would you find me a restaurant in Troy?

Will you find me a restaurant in Troy?

If you could, find me a restaurant in Troy

Chinese restaurants in Troy

A Chinese restaurant in Troy

The Chinese restaurant in Troy

The Chinese restaurants in Troy

Chinese Restaurant in Troy

A Japanese restaurant

Japanese Restaurants

Restaurants that serve Japanese

Restaurants serving Japanese

Restaurants that served Japanese

Restaurants which served Japanese

An Italian restaurant

Italian Restaurants

Restaurants that serve Italian

Restaurants serving Italian

Restaurants that served Italian

Restaurants which served Italian

A German restaurant

American Restaurants

Restaurants that serve steak

Restaurants serving steak

Restaurants that served steak

Restaurants which served steak

An Ethiopian-Italian Fusion restaurant

Brunch Restaurants

Restaurants that serve Spanish food

Restaurants serving American cuisine

Restaurants that served sushi

Restaurants which served ice cream sundaes

Mexican breakfast bistro

Northern Italian cafes

An eatery that serves American food

A hotdog stand in Troy, NY

English pubs that serve soft drinks in Troy, NY

Pizzerias in Albany which served desserts

Eateries that have pizza

Bakeries in New York City

Places to eat near Troy

A Place to eat near Troy that serves Chinese

Diners that have Chinese food

Bars that had pretzels in NYC

Restuarant in Troy that serves sandwiches

sodafountain in Troy that has ice cream

Sicilian pizzeria around Albany, NY

Diner in Latham, NY

Bar near Cohoes that serves hamburgers

Eatery serving wine nearby Troy

Restaurants nearby Troy that serve Chinese

Restaurants of Albany serving desserts

Restaurants by Troy that serve pasta

Restaurants for Troy that have steaks

Restaurants within Troy that serve Mexican food

Restaurants inside Troy, NY

Diners that serve fries around Troy

Restaurants in Albany for pizza.

Restaurants of Italian food in Troy, NY

Diners with ice cream near Cohoes

Restaurants open on Monday

Restaurants open on Mondays

Restaurants open this Monday

Restaurants open next Monday

Restaurants open last Monday

Restaurants that are open on Monday

Restaurants open on Tuesday

Restaurants open on Tuesdays

Restaurants open this Tuesday

Restaurants open next Tuesday

Restaurants open last Tuesday

Restaurants that are open on Tuesday

Restaurants open on Wednesday

Restaurants open on Wednesdays

Restaurants open this Wednesday

Restaurants open next Wednesday

Restaurants open last Wednesday

Restaurants that are open on Wednesday

Restaurants open on Thursday

Restaurants open on Thursdays

Restaurants open this Thursday

Restaurants open next Thursday

Restaurants open last Thursday

Restaurants that are open on Thursday

Restaurants open on Friday

Restaurants open on Fridays

Restaurants open this Friday

Restaurants open next Friday

Restaurants open last Friday

Restaurants that are open on Friday

Restaurants open on Saturday

Restaurants open on Saturday

Restaurants open this Saturday

Restaurants open next Saturday

Restaurants open last Saturday

Restaurants that are open on Saturday

Restaurants open on Sunday

Restaurants open on Sunday

Restaurants open this Sunday

Restaurants open next Sunday

Restaurants open last Sunday

Restaurants that are open on Sunday

Restaurants that were open two days ago

Restaurants that are open two days after today

Restaurants that were open two days before today

Restaurants open today

Restaurants open tomorrow

Restaurants open yesterday

Restaurant open two Mondays ago

Restaurant open two days ago

Restaurant open two weeks ago

Restaurant open two months ago

Restaurant open two years ago

Restaurant open one year ago

Restaurant open a year ago

Restaurant open 1 year ago

Restaurant open this weekend

Restaurant open last weekend

Restaurant open next weekend
Restaurant open on weekends
Restaurants open on Christmas
Restaurants open on Thanksgiving
Restaurants open on Halloween
Restaurant open at noon
Restaurants open at night
Restaurants open at 7:00PM
Restaurants open at 9:00AM
Restaurants open about noon
Restaurants open around midnight
Restaurants open during the weekend
Restaurants for Christmas
Restaurants open in two hours.
Restaurants closed on Mondays
Diners closed Mondays
Diners closed at noon
Restaurants closed around noon
Restaurants closes in two weeks
Diners closed for Christmas
Diners closed at midnight.
Restaurants closed about 10:00 PM
Restaurants rated 0 stars
Restaurants rated 1 stars
Restaurants rated 2 stars
Restaurants rated 3 stars
Restaurants rated 4 stars
Restaurants rated 5 stars
Restaurants rated zero stars
Restaurants rated one stars
Restaurants rated two stars

Restaurants rated three stars

Restaurants rated four stars

Restaurants rated five stars

0 star restaurants

1 star restaurants

2 star restaurants

3 star restaurants

4 star restaurants

5 star restaurants

Diners not in Latham

Diners that are not in Latham

Diners not serving pizza

Restaurants not inside Troy

Bars except in Troy

Diners in Troy or Albany that serve breakfast

Chinese restaurants in both Albany and Cohoes

Restaurants in Troy that have tacos and pizza

Restaurants in Albany that serve sushi or noodles

Fast food or diners in Albany serving fries

Bars or restaurants in Albany serving beer and wine

Troy’s diners that serve hamburgers

Albany’s Chinese restaurants

Cohoes’ Eatery serving wine

Large restaurants in Troy

Troy's large restaurants

Small restaurants in Troy

Middle-sized restaurants in Albany

Big restaurants in Cohoes

Mdm diners in Troy, NY

Restaurants in Troy rated highly

Diners rated highly in Troy

Eateries in Albany that are rated highly

Search Chinese restaurants

Serving Mexican food

Not serving meat

Search for Chinese restaurants

Restaurants rated two stars

Restaurants rated highly

Food Court Restaurant that serves Pizza

Gas Station Fast-Food Place with Ice Cream

26. Appendix 3: Sports Domain Corpus

leading rusher day 1 last 5 years

team josh freeman has the most yards against?

player had more than 100 yards in interception returns

player has more than 100 yards in interception returns in 2012

joe flacco passes for over 300 yards or a td

scored the most points against the steelers in 2008

yards per game did adrian peterson average last october

ravens players against the afc south

the highest touchdown scoring quarterback in the 2000s

quarterback the steelers sacked the most in december

team brady passed for the most yards against

yards for ray rice against the steelers during the 2010 regular season

player caught the most passes from brady

tight end caught the most passes from brady

passes did brady throw to welker

defenses playing against the patriots

defenses playing against tom brady

defenses playing tom brady

defenses is playing tom brady

offensive backfield players on the giants

2003-2010 steelers except 2004-2005

december ravens against tim tebow's team

any quarterback except for tom brady

2011 south afc passing plus bears

quarterbacks, tom brady excepted

anyone on the broncos except for tim tebow total 2011 tds

everyone on the broncos with the exception of tim tebow

tim tebow and peyton manning do better versus the packers

jacksonville and dallas that tim tebow does best against

tim tebow and peyton manning does better versus the packers

whether eli or peyton manning plays better against the giants
top completion percentage
adrian peterson carries for each touchdown
is peterson better on grass or astroturf
peterson grass vs. manning grass
year is on record for the fewest giants tds
player, tom brady or tim tebow, passes longer
peterson grass vs. manning grass not manningham
yards for non-giants who have more than 10 touchdowns
anyone who can throw other than tebow, manning, brady, tynes, or rodgers
players who rushed for more than a hundred yards against the steelers last year
tight ends play for the giants right now
scored the most points against the steelers in 2008
200 yard games did adrian peterson have
passing and rushing touchdowns manning had last year
rushing or passing touchdowns manning had last year
some of the years that tom brady was with the patriots
yards for non-giants who have more than 30 fantasy points
better of 2010 brady and 2011 brady passing
quarterback started for the steelers last year against the giants
month does peterson run for the most yards: december or january
pats or bears or giants had the most 2011 tds
of the bills/steelers/pats had the most points in the 2010 season
bills/steelers games the bills won
quarterback or wide receiever threw the fewest yards in the afc east
best month for td passes tom brady
buffalo bills september record the past 5 years
fumbles for tom brady
eli how many fantasy points last season
have any yards last year
quarterback had the most yards against the steelers in december

quarterback passed the most yards against the steelers
defense had the fewest scores against them in 2011
running back scored the most touchdowns in october
ray rice against the afc north in decembers
quarter back threw for the most yards against the steelers
running back scored more than 10 touchdowns
quarterbacks threw 10 touchdowns
yards bettis rushed for against the ravens in 2005
defense had the most sacks last year
quarterback scored more than two touchdowns against steelers
best year rushing for nfc west in the 80s
best year rushing for nfc west
best year for nfc west
player rushed the most in a single year
the new england pats offense yards
running back caught more than 50 passes in 2012
record number career pass yards for a giant
season-long passing yards by brady
passer rating for afc players who have more than 2000 yards
player scored more touchdowns than tom brady
quarterback threw for more yards than tom brady
yards for ray rice at heinz field
yards for ray rice against the steelers at heinz field
astroturf away games were won by the patriots between 2008 and 2010
ravens last decade
giants and new england two years ago
least yards rushed in a year for giants
ny wins against pats
player has the most rushing yards of all time
2010 tom brady best of passing or rushing yards
2011 tom brady best of passing or rushing yards

2002 all stats but passing
texan kickers had a higher attempt percentage than giants in the last ten years
domed, undomed stadium games featuring eli manning
yards for non-giants
passing for anyone but tim tebow on his 2011 team
tom brady in domes vs outside
tom brady vs domes
tom brady on grass vs astroturf
tom brady vs grass
touchdowns were made by pittsburgh steelers in 2010?
tom brady total 2011 tds
tom brady total 2011 yards
manning is the more valuable player
raiders passing numbers
tom brady or eli manning only against ravens
just quarterbacks passing against 2011 ravens
1990's steelers passing
2011 afc passing except jaguars
steelers passing in the 1990s
december afc passing except jaguars
eagles between 2001 and 2004 passing numbers
top rushing or passing touchdowns
top rushing or passing
bears or packers
bears or packers rushing
steelers and packers passing
how the steelers did against the patriots for 2001 season
late fall games played by the bills
players rushed for the most yards in week 2 of 1963
players rushed for the most yards in week 2 of 1945
touchdown passes were thrown by y a tittle in 1963?

players for the steelers threw the ball more than 100 yards in 2002
who threw the best for the bills last year
surface the steelers did the best on in 2011
quarterbacks in the afc west the end of the 2002 season
quarterbacks in the afc east for passing the end of the 2002 season
bills quarterbacks played better for the 2010 or 2011 season
quarterback threw the most touchdown passes in 1963?
touchdowns from the top guys in 2010
players stood out in the 2011 season
times tom brady muffed the ball in 2012
games between peyton manning and eli manning in 2010
player lead the league in rushing touchdowns
games between bears and packers in 2012
games that tim tebow fumbled in for the 2010 season
defense did the best against tom brady
2012 passing not december
who has the most yards rushing against the nfc north
yardage gained by giants in 2010
steelers wins against pats
yardage teams in afc
colts defense yards, passing
who had the least touchdowns in the playoffs the end of last season
december ravens in years in the 2000s
1990s player with most yards
2010s afc teams player with most yards
players scored no fantasy points last year
player rushed for the most yards against the steelers in 2008
quarterback throws for 5000 yards the last 3 years
bettis
player is the tight end for the steelers
jones catches

yards for ray rice against the steelers in 2010
peyton manning rushing yards in 2007
passing yards for manning in the last five years
player scored the most points against the steelers in 1979
yards bettis rushed for against the dolphins between 2003 and 2005
kickers did the best in the afc west last year
giants games
2001 team except the steelers had 5 touchdowns in a game
giant rushed for more than 200 yards
receives the giants had in 2010
tight ends play for the giants now
giant rushed for 200 yards
who had the most 2 point conversions
most points not kickers
touchdowns tom brady scores against the steelers in november
touchdowns tom brady scored against the steelers in november
tim tebow in 1995
tim tebow in 1945
player had the most tackles against the patriots
running back is best against the packers in 2011 and 2012
steelers schedule last year
player has the most tds in decembers
running backs against the afc south
ravens against the afc south
tds quarterbacks had on the road
best field goal kicker on the road
team had most passes to te in 2011 and 2012
running back has more than 2 tds in a game
most points non kickers
yards rushing for calvin johnson in 2011 and 2012
passing tds for tom brady against afc

player had the most 300 yard games
player scored the most points against the steelers in 1973
tight end made the most 2 point conversions
adrian peterson in novembers
top rushers for 2011 and 2012
team rushed for the most yards against the steelers last year
quarterbacks averaged the fewest interceptions in 2011 and 2012
player ran for the most yards against the steelers in 1975
what was the score of the 1975 steelers vs. bills game?
adrian peterson scores
yards did adrian peterson average last october
yards did adrian peterson get
team has the best quarterback
how many yards did foster have in 2012
injuries for nfl
receptions adrian peterson has in road games
yards adrian peterson rushed for during the last 12 months
average adrian peterson rushing yards for the last five years
running back had more than 2 tds in a game
passing yards in 2012 for nate washington
recieving yards did t.y. hilton have in2012
nick foles passing stats for 2012
quarterback threw four interceptions
running back had the most 100-yard games in 2012
yards bge had 2012
injuries for manning
player got the most yards last year against denver
who will have the best fantasy season
rob gronkowski passing yards
spiller receptions and receiving tds 2011 and 2012
josh freeman throws for 300 yards in one game

how many interceptions has tom brady thrown
player got 200 yards rushing last year
eagles player has most receptions in 2012
joe flacco passes for over 300 yards
yards adrian peterson rushed for over the last 12 months
quarterback has the best touchdown interception ratio
fumbles does calvin johnson have in 4th quarters
fumbles does calvin johnson have in road games
yards adrian peterson rushed for between october and december
games steelers lost in 1945
rushed for the most yards against the steelers in 2008
player scored the most points against the steelers in 1979 during home games
2011 bills away astroturf contests
2002 bills away astroturf contests
afc players that ran for at least 1000 yards
the the bears
the top 2 or 3 qbs in the afc east last year
TOUCHDOWNS for 2011 tim tebow
team tebow had the most touchdowns for
adrian peterson last game rushing
quarterback is the best in the afc period
defense racked up the most sacks last year
player had more than 100 yards rushing in a single game
leading rusher game 1 last 3 years
running back caught more than 50 passes in 2011
quarterback had the least interceptions per game
defense allowed the most red zone touchdowns in 2012
top 2 or 3 qbs in the afc east last year
julio jones receiving yards in 2012
calvin johnson receiving yards in 2012
t.y. hilton 2012

nate washington yards
calvin johnson yards
nate washington
receiving yards in 2012 for nate washington
passing yards in 2012 for nate washignton
hilton receiving yards
recieving yards did t.y. hilton have in 2012
t.y. hilton
hilton
nate washington in 2008
nathaniel washington
receiving yards nate washington had in 2012
donnie avery receiving yards in 2012
nate washington receiving stats
nate washington receiving yards
receiving yards natec washington had in 2012
nate washington recieving stats
recieving yards
t. y. hilton receiving yards in 2012
quarterback threw four interceptions
quarterback threw 4 interceptions
quarterback threw for at least 15 touchdowns in 2012
player had the most rushing yards in 2009
how many tds did adrian peterson have in 2010?
running back had the most 96 yard games
running back had the most 100 yard games
benjarvus green ellis
green ellis
yards bge had 2012
brandon myers 2012
jackson

injuries for manning
anquan boldin yards receiving
vincent jackson yards receiving
running back had the most 100-yard games in 2012
player had the most 100-yard games in 2012
player has more than 400 yards rushing
yards did you have
tackles did timmons have
tackles did you have
tackles did you get
yards peyton manning had in 2010
player had the most yards last year against denver
player got the most yards last year against denver
running back had the best rookie season
injuries for patriots
yards tim tebow had last season
frank gore rushing yards
reggie bush rushing yards
who will have the best fantasy season
touchdowns for lesean mccoy
had the most td receptions in 2012
buffalo bills scoring tds against ravens
buffalo bills scoring tds against ravens for the last five years
list buffalo bills scoring tds against ravens
which buffalo bills has most tds against ravens
tom brady against the bills
team with the most rushing touchdowns
rob gronkowski passing yards
alex smith qb rating
aaron rodgers qb rating
what was joe montana's career passing total

49ers quarterback has had the best career qb rating
lynch
injuries for seahawks
player had most rushing yards
player caught most receptions in 2012
players threw for more than 100 yards in a game last year
quarterback averaged the least interceptions
quarterback had the least interceptions in 2012
which quarterback had the least interceptions in 2012
cj spiller receptions 2011
dez bryant receptions 2011
games calvin johnson played last year
brett favre
players rushed for over 500 yards last year
alfred morris in 2012
dez bryant receptions and receiving tds 2011 and 2012
spiller receptions 2011 and 2012
dez bryant receptions 2011 and 2012
c.j. spiller receptions 2011 and 2012
cj spiller receptions 2011 and 2012
interceptions tom brady threw
interceptions tom brady
interceptions does tom brady have
player has the most interceptions
how many interceptions josh freeman
how many interceptions does tom brady have
peyton manning rushing yards
eli manning rushing yards
got more than 200 yards rushing last year
had more than 200 yards rushing last year
player rushed for more than 200 yards

rushed for more than 100 yards last year
tom brady in 2012
rg3
rg iii
got 200 yards rushing last year
robert griffen
qb had the most touchdowns in 2012
adrian peterson scores a touchdown
injuries for arian foster
quarterback is best in december
player has the most rushing yards in 2012
tim tebow rushes for 100 yards
chris johnson yards
arian foster yards
adrian peterson rushing yards for the last five years
adrian
injuries for arian foster
tim tebow 2011
adrian peterson scores a touchdown
tom brady 2007
injuries for rgiii
aaron rodgers yards
cam newton yards
yards adrian peterson rushed for the last 12 months
tom brady tds
nfc qbs tds
andy dalton tds
andrew luck tds
cam newton rushing yards
rgiii rushing yards
touchdowns for tom brady

running back rushed for more than 50 yards against the steelers in 2008
running back rushed for more than 100 yards against the steelers in 2008
player has more than 100 yards
adrian peterson in december
adrian peterson against the steelers
player ran for the most yards against the steelers
yards adrian peterson rushed for in october
yards adrian peterson rushed for last year
yards did adrian peterson average
adrian peterson rushing yards for the last 5 years
players ran for 150 yards
tom brady throws a touchdown
brady
yards for brady
yards brady threw for
score of the 1975 steelers vs. bills game?
quarterbacks averaged the fewest interceptions in 2012
adrian peterson in 2008
adrian peterson in 2005
games steelers lost in october 1965
games steelers lost in 1965
passing tds for matt shaub
games steelers won in 1945
steelers schedule in 1945
steelers schedule in 1932
steelers schedule
tim tebow in 1970
passing tds for matt shaub
top rushers for 2011 and 2012
passing tds in 2011 for stafford
passing tds for stafford

aaron rodgers tds
drew brees tds
adrian peterson in november
running backs with most 150 yard games
top performances
foster yards
2011 most points
2012 most points
2010 top rushers
2012 top rushers
yards rushing for calvin johnson in 2012
peyton manning yards
ben roethlisburger yards
peyton manning yards
arian foster tds
ben roethlisburger yards
how fumbles steelers forced
how many interceptions in 2012 for chicago bears
wallace
touchdowns tom brady scored against the steelers in december
yards bettis rushed for against the dolphins between 2005 and 2008
players rushed for the most yards against the steelers in 2008
yards adrian peterson rushed for against the bears in 2010
adrian peterson against the bears in 2011
touchdowns tom brady scores against the steelers in december
yards bettis rest for against the dolphins between 2005 and 2008
tom brady rushing yards in 2007
peyton manning rushing yards in 2008
tom brady rushing yards in 2008
passing yards for manning
adrian peterson in 2011 on turf

adrian peterson in 2008 on turf
catches for tim tebow
yards for ray rice against the steelers
smith catches
tebow catches
adrian peterson on astroturf
adrian peterson on turf
jerome bettis
player rushed for the most yards against the steelers
players who rushed for more than 100 yards against the steelers
players who rushed for more than 100 yards against the steelers last year
quarterback throws for 5000 yards in a season
yards bettis ran for in 2005
yards for non-giants who have more than 30 fantasy points
passer rating for afc players who have more than 30 fantasy points
peyton manning how many fantasy points last season
steelers yards, passing
top passing in 2011
best tds
worst giants tds
1990's ravens passing
team has the best defensive roster
defense did the best against tom brady during his career
y. a. tittle
tom brady or eli manning only
times tom brady muffed the ball in 2010
games there were that tim tebow fumbled in for the 2010 season
games between payton manning and eli manning there were in 2011
receiving yards calvin johnson had last year
fumbles the giants forced in 2002 in games against the ravens
who threw the best for the bills in the last decade

adrian peterson rushes for 100 yards
times tim tebow fumbled in 2010
afc east qbs excepting bills
buffalo bills yards
games tom brady played against the jets
tom brady on grass for his whole career
games tom brady has played against the jets
rushing yards for tom brady
christian ponder september
vikings september
tom brady on grass
the steelers against the ravens in october
the steelers in october
record for san fran touchdowns
record for frisco touchdowns
2003 bills away astroturf contests
matt schaub yards
matt yards
alfred morris yards
marshawn lynch yards
nfc rushing leaders 2012
blaine gabbard
blain gabbard
Rob gronkowski
peyton manning
tom brady
matt ryan tds
ray rice yards
nfc west scores
ravens
adrian peterson

quarter back does best against the steelers in december

packers

how many yards peterson rushed for last year

quarterback scored more than two touchdowns against the steelers in 2005

yards peterson rushed for against the bears in 2012

yards peterson rushed for against the bears in 2011

jerome bettis in december 2005

jerome bettis in december

yards peterson rushed for against the afc north

yards bettis rushed for against the steelers

top performers

top quarterback in the early 90s

tom brady passing

peyton manning passing

field goals for steelers

afc north passing

june rushes for the pats

tds the pats had against the steelers in 2009

tim tebow

ej manuel

balitmore's team from january

afc rushing leaders 2012

show is the quarterback for the steelers

defense had the fewest scores Gainst them in 2011

nfc west passing at home

nfc east passing at home

vikings defense depth chart

vikings offense depth chart.

dal passing

sea passing

team did best against broncs in 2010

packers depth chart
field goals the buffalo bill made while away in 2011
quarterbacks of nfc north
nfc north touchdowns
nfc north passing
quarterback of minnesota vikings
nfc rushing
who is the current leader in yards
which quarterback threw the most touchdowns
quarterback with the most touchdowns
who threw the fewest interceptions
which quarterback threw the fewest interceptions
nfl
nfc
nfc north
the current leader in yards
rushing stats
top receivers
recievers
rushers
pass stats
rushing leaders
adrian peterson
adrian peterson carries per touchdown
adrian peterson touchdowns per carry
adrian peterson rushing
adrian peterson yards
adrian peterson rushing yards
adrian peterson tds
adrian peterson touchdowns
adrian peterson yards per carry

minnesota vikings
top passing
top passers
top rushers
who is the current leader in touchdowns
ponder against the packers
peterson against nfc north
ponder against nfc
ponder against aaron rodgers
vikings against nfc north
vikings against packers
vikings against aaron rodgers
vikings against nfc
which running back caught the most touchdowns
which player caught the most touchdowns
nfc passing
which running back ran the most touchdowns
which player ran the most touchdowns
which player threw the most touchdowns
nfc north against nfc
nfc north against green bay packers
nfc north against aaron rodgers
nfc north against nfc north
top rushers against the packers
top passers against the packers
quarterback of the vikings against the packers
top rushers against aaron rodgers
top passing against aaron rodgers
quarterback of the vikings against aaron rodgers
top rushers against nfc
top passing against nfc

quarterback of the vikings against nfc
top rushers against nfc north
quarterback of the vikings against nfc north
top quarterback against the packers
top quarterback against nfc north
top quarterback against nfc
top quarterback against aaron rodgers
vikings against packers playoffs
vikings playoffs
adrian peterson against green bay playoffs
adrian peterson playoffs
adrian peterson
nfc north playoffs
top rushers playoffs
top rushers in the playoffs
top rushers against the packers in the playoffs
rushing against nfc north
rushing against aaron rodgers
who had the most touchdowns in the playoffs
nfc north week 17
nfc north week 1
who had the most interceptions in december
top passers week 17
who had the most touchdowns in week 17
vikings week 17
vikings quarterback week 17
christian ponder week 17
top players week 17
top player playoffs
top player
top player against the packers

nfc north in december

standings

vikings standings

nfc north standings

top performer

top performer week 17

top performer playoffs

top performance

top performance week 17

nfc north scores in september

nfc north scores in december

nfc north schedule in september

nfc north schedule in december

december scores

december schedule

scores

packers scores

top rushing performance in 2012

top performance playoff

vikings record at home

vikings record away

running backs for the steelers

quarterback for the steelers in 2010

who is the quarterback for the steelers

new orleans saints

record on the road

quarterback of the vikings

who has more than 35 touchdowns

more than 200 yards

how many yards did dennis dixon pass for in 2010.

week 1

week 17
who is the current leader in receiving touchdowns
who is the current leader in receiving yards
nfc north september
nfc north this year
nfc north last season
adrian peterson december
vikings december
vikings rushing week 17
vikings 2012
vikings rushing 2012
show who is the quarterback for the steelers
who threw the most touchdowns
who rushed for more than 1000 yards last year
who rushed for more than 2000 yards last season?
receivers who caught more than 100 passes in 2008
who threw for More than 1000 yards in 2008
who rushed for more than 1000 yards in 2008
how many yards bettis rushed for in 2004
which quarterbacks passed for more than 3000 yards in 2008
wide receivers who caught more than 5 touchdown passes in 2009
which defense had the most yards per carry at home
packers rushing defense
packers against the pass at home
packers against the pass
best defense against the run
which team gave up the fewest touchdowns
which defense had the most yards per carry
which defense had the most touchdowns
which defense had the most yards
which defense had the most interceptions

worst defense
packers defense
who rushes for more than 100 yards against the patriots
how many yards does peterson run for in december
receivers with more than 2 touchdowns in a game last year
top rush
highest rated qb
qb with the highest rating
top rated passer
the player who scored the most points in 2009.
Running back is best against the packers
defense forced the most fumbles
player has the most TDs
player leads in rushing TDs
team has the most yards per carry
team has the most rushing yards against the NFC West
manning
defense with the most interceptions
Adrian Peterson at home
Tom Brady against the AFC East
Baltimore Ravens in December
players with more than 4000 yards passing
players that caught more than 3 TDs
TDs the packers had on the road
touchdowns were scored by the packers
how many yards peterson rushed for
how many yards peterson rushed for last season
most 200 yard rushing games
Yards for Manning
Yards for the Tampa Bay Defense
Drew Brees passing yards

RGIII passing yards
Steelers defense yards
Alfred Morris touchdowns
top rushers touchdowns
Receptions for Marshawn Lynch
fumbles the steelers forced in december
Passer rating for Tom Brady
who made the most field goals
who attempted the most field goals
field goal percentage
who had the highest field goal percentage
team had the highest field goal percentage
xps
field goal attempts
who had the most field goal attempts
steelers field goal attempts
kicking
adrian peterson fumbles by game
defense is best against the run
catches calvin johnson had last year
points the bills scored at home
yards drew brees passed for
top passer rating in december
top quarterback rating
defense is best
player scored more than three touchdowns against the steelers
how many fumbles for michael vick in 2011
how many yards for calvin johnson in 2011
ben roethlisberger tds
alfred morris scores a td
joe flacco yards

running back had the best rushing yards against the packers

blaine gabbert

running back had the best rushing game against the packers

fewest carries per touchdown

bears kick a fg

peyton manning throws a touchdown

yards for tom brady

RGIII

arian foster receiving yards

qb had the most yards in 2011

quarterback had the most yards against the steelers

quarterback passed the most yards against the steelers in december

yards for adrian peterson in 2008

running backs ran more than 1000 yards last year

yards for brady against the steelers

yards for peterson against the steelers

yards for foster against the steelers

peyton manning throws a TD

ravens points

steelers points

quarterback had the highest passer rating in December

tight end caught the most passes last year

brady against the steelers

player rushed for the most yards last year

player rushed for the most yards against the steelers last year.

yards drew brees passed

how many yards drew brees passed for.

player has more than 400 yards

chicago defense interceptions 2011

chicago defense

quarterbacks in december

defense had the most sacks
yards
yards for calvin johnson last year
top quarterback in the afc
best running back in december
tom brady sacks
ray rice against the steelers
quarterback passed for the most yards in away games
running back rushed for the most yards against the steelers last year
who had the most 200 yard rushing games
the number of yards bettis rushed for in 2007
how many yards bettis rushed for in 2007
running backs who rush for more than 100 yards against the patriots
quarterback does the best against the steelers
running back does the best in december
defense had the fewest points against them in 2011
defense had the fewest scores against them
balls rice fumbled
times rice fumbled
running back to start this week
steelers and packers
yards and touchdowns
top rushing and passing
top rushers and passers
top yards and touchdowns
packers roster
who plays for the saints
top quarterback inside their division
top quarterback against their division
times ray rice threw an interception last year
qb had the best qb ranking inside their division

tom brady and peyton manning
tom brady and peyton manning passing
ray rice against the afc north in december
running back scored the most touchdowns in december
passes were thrown against the steelers
adrian peterson against the bears
scores for the steelers in december
defense is best against the rush
fumbles for the steelers quarterback
have any yards last year
quarterback threw for the most yards against the steelers
packers outside
top rushers outside
packers on grass
adrian peterson on grass
adrian peterson on artificial turf
packers on artificial turf
top rushers on artificial turf
top rushers on grass
quarter back threw for the most yards against the steelers
top rushers in domes
players scored more than 10 touchdowns
quarterbacks threw more than 10 touchdowns
yards peterson rushed for against the bears
yards peterson rushed for
yards bettis rushed for against the ravens
how many yards did bettis rush for against the ravens in 2005
running back had the most touchdowns against the steelers in 2010
running back had the most touchdowns against the steelers
players scored a touchdown against the steelers
player rushed for more than 100 yards against the steelers

player rushed for more than 100 yards against the steelers in 2012
player scored more than 100 yards against the steelers in 2012
defense recorded the most sacks last year
yards peterson rushed for against the afc north in 2011
top rushers in the last two years
adrian peterson in the last 2 years
is adrian peterson injured
touchdowns adrian peterson averages
Average yards for Brady in 2012
yards peterson averaged against the bears in 2012
adrian peterson status
injuries december
injury report
adrian peterson last game
nfc west last game
adrian peterson last season
who rushed for more than 2000 yards last year?
quarterback scored more than two touchdowns against the steelers
quarterback scored more than two touchdowns against the steelers in 2005 and 2008
defense is dominant in the red-zone
saints in decembers
adrian peterson in decembers
top rushers on mondays
who does the best on monday
quarter back did best against the steelers in decembers
top rushers in the late 90s
adrian peterson late october
adrian peterson early december
adrian peterson mid december
quarterback is the best of the early 90's
defense is best against the rush in the first half

defense is best against the rush in the third quarter
team led in touchdowns october last year
interceptions for steelers and giants 2011
afc east passing
yards for patriots quarterbacks last year
how many passing yards for joe montana
a. peterson in 2012
top quarterback in the early 00s
baltimore's team from january
the new england pats best yards
player in nfc west led in rushing this past year
the pats in away dome stadiums
doesn't have any yards last year
players rushed less than 20 yards in a game in januaries
worst years yards
tds the pats had against the steelers in 2010
september rushes for the pats
doesn't have more than 4 yards per carry last year
doesn't have more than 90 yards last year
team has the best wide receiver
average rush attempts for peyton manning in 2010
quarterbacks do the best in december
player made the most passes
team did best against broncos in 2010
teams had 2 touchdowns in a game last december
vikings offensive depth chart
vikings special teams depth chart
nfc rushing leaders 2012
passing and rushing tds for steelers and patriots
passing efficiency for steelers offense
steelers on grass last 3 years

afc east players who threw for more than 300 yards in 2010
individual offense leaders from 2012
player singlehandedly had the most touchdowns in 2010
field goals the buffalo bills made while away in 2011
who does manning start for
vikings defensive depth chart
a good player for touchdowns at home
away games played by the patriots in 2000
the worst player for pats passing 2010 away games
record for san fran touchdowns in a game
record for san fran touchdowns
players rushed for the most yards in week 2
the steelers against the ravens in november
games the ravens won from 2000 to 2005
games the ravens won from 2000 to 2005 against buffalo
yards for lead patriots quarterback last year
interceptions steelers had while on the road
Players rushed for 1000 yards or more last season
touchdowns the pats had during home games in fall
player from afc west had the most away touchdowns last season
player had the most touchdowns in 2010 by himself
passing attempts for manning against the steelers
games tom brady played against the jets in 2006
preseason games tom brady played against the jets in 2006
tom brady career touchdowns against the jets
tom brady touchdowns away for his whole career
tom brady touchdowns for his whole career
tom brady on grass for his career
tom brady at home for 2008-2011
tom brady on grass for 2008-2011
broncos other than manning

touchdowns for manning not peyton
tom brady or peyton manning
top three teams for 2011 rushing
top 3 players for touchdowns
record year for ravens wins
broncos other than manning had more than 300 yards passing
patriots except tebow
rushing nfc west not packers
most touchdowns not qb
ravens in december yards
fumbles the colts forced in 2002 in games against the ravens
quarterback threw the fewest yards in the afc east
defense made the most fumbles and interceptions
2011 afc south passing not minnesota
2011 afc passing except afc south
2010 quarterbacks (no bills) had more than 100 yards passing
the defense that forced the most fumbles and completed the most interceptions
team the steelers played in the first week of november
qbs playing the steelers in the 1990s
steelers had no rushing tds in sept 2011
steelers players had no rushing tds in sept 2011