GENERATION AND EVALUATION OF LINKED DATA DERIVED FROM INFORMATION EXTRACTION METHODOLOGIES

Ian Gross

Submitted in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

Approved by:
Dr. Deborah L. McGuinness, Chair
Dr. Heng Ji
Dr. James A. Hendler

Department of Computer Science
Rensselaer Polytechnic Institute
Troy, New York

[May 2018]
Submitted March 2018

Contents

List of Tables

List of Figures

ACKNOWLEDGMENT

First of all, I would like thank my thesis advisor, Professor Deborah McGuinness, for her support throughout my research at Rensselaer and guidance towards completion of my thesis. My introduction to the world of the semantic web would not be possible without Professor McGuinness, for that I am deeply grateful.

Secondly, I would like all the individuals that helped with the completion of this work. I would like to thank James McCusker for providing the inspiration of my work, his support within WhyIs, and guidance throughout the workflow implementation process. I would like to thank Sabbir Rashid for providing assistance and mentorship throughout my time at the Tetherless World Constellation. I would like to thank Professor Heng Ji and Lifu Huang from the BLENDER lab for their support in relation to the Liberal Event Extraction software.

I would like thank all my committee members, Dr. Deborah McGuinness, Dr. Heng Ji and Dr. James A. Hendler. Their feedback and attention to my thesis work are greatly appreciated.

I would like to thank all the members of envirohealth team and the Tetherless World Constellation at RPI for their support throughout my research career.

Finally, I extend my deepest gratitude towards my family and all the friends I have made throughout my time at RPI. Their support, motivation, and encouragement were indispensable towards pursing my passion and completing my degree.

ABSTRACT

Many approaches currently exist in the pursuit of knowledge representation from unstructured documents. With the extensive amount of information available, the need to derive a common meaning behind data has become more prevalent than ever before. In this thesis, we develop a workflow to support knowledge extraction and representation of data from the biomedical domain. To determine the hidden meaning behind text, we utilize popular Information Extraction methods to extract events from biomedical papers and store the output from these methods into a combined textual object and annotation representation format. This data is converted into an RDF Graph format based on groundings to scientific ontologies and provenance semantics.

By employing the extensible knowledge graph curation and analysis platform called Whyis, we can validate the constructed interactions discovered by our workflow in comparison to interactions stated in a domain-specific assertion database. Assessment of this data flow showcase the associated benefits, difficulties, precision, and potential usages alongside established ontologies. The data workflow implemented in this study provides an innovative approach for knowledge discovery, connectivity, and curation of information using a linked data methodology and assertion-based evaluation criteria.

Chapter 1
INTRODUCTION

1.1 Motivation

Over this past decade, we have seen a massive increase in the amount of data available over the Internet. Traditional literature is more accessible and we produce more information since the introduction of the Web 2.0. With this large influx of user created content and availability of this information, the need to condense and extract knowledge has become more important than ever before.

Natural Language Processing has made great strides over the past few decades to automatically extract meaningful information from document collections [1]. This task is better known as Information Extraction, which takes natural language as input and identifies various entities and events described throughout a document [2]. Although this extracted knowledge may be useful to us in the short-term, it is necessary to introduce connectedness into our data. The semantic web seeks to build a web of data that enable computers to interact in a common format [3]. Semantics provide a foundation to assert knowledge found in documents and provide provenance pertaining that a particular document.

The primary focus of this study is centered around knowledge extraction from traditional literature and connecting this knowledge to linked data. Introducing semantics into our knowledge supports further knowledge curation tasks such as inferencing and reasoning, allowing for new information to arise that wasn't accessible previously. Using this ideology, we determine the viability of this process for two Information Extraction tools by implementing each tool's data workflow. Additionally, we explore the precision of our primary workflow for the biomedical domain and compare results against manual assertion databases.

1.2 Procedure

The workflow for this thesis can be broken down into three major components: information extraction, linked data embeddings, and evaluation. The information extraction methodologies utilized are ODIN [4] and Liberal Event Extraction [5]. Both information extraction methodologies are both domain-independent and require no training data to extract

events from documents. We will explore more about these methodologies in the Background section.

We applied this workflow to the biomedical domain, specifically those describing protein interactions. The biomedical domain was chosen due to its time-intensive nature to determine essential events located in these documents. The information extraction within the biomedical domain has been explored in various recent studies, but they all fail to realize the potential in rapid adaptation of a biomedical information exaction methodology for knowledge management and discovery. To utilize this knowledge, we need to convert the output of these information extraction methodologies to linked data. Each methodology is able to represent its data in a structured frame representation [6]. If we can understand the semantics of each frame, grounding to external linked data references (otherwise known as an ontology) can be determined.

What is the significance of having our extracted knowledge in a linked data format and how do we evaluate output of our workflow? In each document, there can be some expected knowledge located within this document. By utilizing a factual index of protein interaction records, we can determine if our workflow is able to distinguish these facts without any prior knowledge of these records. Additional significance can be obtained by formatting our linked data as Nanopublications [7], which is more commonly described as RDF graphs. Furthermore, we utilize a new nano-scale knowledge graph publishing, management, and analysis framework called Whyis [8] to support inferencing of our created Nanopublications.

1.3 Contributions

In this thesis, we develop an innovative approach to discovery, connectivity, and curation of knowledge from two different information extraction methodologies. The majority of our contribution is the creation of a knowledge extraction linked data workflow. Some portions of the workflow are built by scratch, while others include pre-existing software that must be connected to fit into a larger picture. The rest of our contributions come in the form of the workflow evaluation. Our evaluation approach is novel in comparison to similar studies using information extraction and indicates several implications for necessary components of all provenance-based knowledge management systems. Finally, we explore the potential impact and changes necessary of this workflow to apply our work to concrete use case validation scenarios.

Chapter 2
BACKGROUND

2.1 Information Extraction

2.1.1 Overview

Natural Language Processing (NLP) is a discipline in which a computer attempts to understand, analyze and generate human recognizable speech or text. The foundations of this technology are primarily associated within computer science, artificial intelligence, mathematics and linguistics. Natural Language Processing relies on formal modeling of natural language based on phonology, phonetics, morphology, syntax, semantics, pragmatics and discourse [6]. In order to capture this knowledge, several models such as probabilistic, formal rules, and logic based systems have been developed. Some common areas for natural language based systems include machine translation, question answering, word sense disambiguation and information extraction.

Information Extraction is the formal process of interpreting information from unstructured or semi-structured documents [6]. Information Extraction is a subtask of Natural Language Processing, utilizing methods from the field of study to extract implied meaning from provided input. The typical information extraction tasks consists of filling a template, or frame with slots, that need to be filled in with the material from the provided text [6]. The standard evaluation criteria for an information system can be borrowed from the field of information retrieval: precision, recall, fallout and F-measure [6]. These measures are often used as the standard method for evaluation within natural language processing field, pertaining primarily to those processes that have an expected outcome and can be measured based on its overall correctness.

Two common representations for information extraction output are entities and events. A named entity is a word or group of consecutive words found in a sentence that represent concepts such as a person, location, organization or geopolitical entity [9]. Named entity recognition is utilized in a countless number of information extraction tasks for various problems throughout NLP. For example, Jin Guang Zheng developed an entity linking approach to extracting entity mentions from unstructured biomedical literature and grounding them

in terms described in a knowledge base [10].

An event can be described a by change based on context. Each event consists of a trigger word(s) and the participants involved in the event (arguments) [5]. An event mention is used to describe the trigger(s) and the arguments as a whole phrase. This pattern is known as syntactic dependency representation and is commonly used to represent events [4]. To demonstrate this concept, take the following sentence as an example. The bold word represents the trigger, the underlined words represent the arguments, and each argument has some role in relation to the trigger (not labeled in diagram):

S100A4 ***promotes*** *angiogenesis in vivo by preventing the anti-angiogenic effect of THBS1.*

Entity and event recognition can either be determined utilizing supervised or unsupervised methods. Supervised learning assumes that the provided data has an associated labeling or context. Unsupervised learning has no initial labeling for the initial data and must be annotated. To extract meaning from a document, one must utilize an unsupervised learning method to identify these labellings. Various common labeling implementations exist, such as part-of-speech tagging and clustering.

In summary, Natural Language Processing is utilized to allow a machine to intelligently parse text from a human source. The rapid development of the World-Wide Web opens many possibilities for applications to be built on top of Natural Language Processing. We apply unsupervised information extraction tools to extract events from a document. By classifying knowledge into frames, we are able to assign ontological representations to each frame and apply them in knowledge management or discovery applications.

2.1.2 Open Information Extraction Software

There is a wide variety of open information extraction software currently available, which utilizes unlabeled training data. University of Washington's 'Know It All' team created a popular Open Information Extraction system to parse relations (triples) from sentences called OLLIE [11]. OLLIE accomplishes this using Semantic Role Labeling and open pattern matching templates. General Architecture for Text Engineering (GATE) provides a graphical evaluation infrastructure for language processing through language engineering modules [12]. NLP distribution packages such as The Standford CoreNLP Toolkit [13] and the Natural Language Toolkit (NLTK) [14] offer packages to support tasks such as tokenization, named

entity recognition, coreference and dependency extraction.

For this thesis we chose two open information extraction tools to design event extraction workflows, with a focus on event extraction. The first tool is the Liberal Event Extraction paradigm, developed by Lifu Huang at Rensselaer Polytechnic Institute [5]. Liberal Event Extraction takes Abstract Meaning Representation (AMR) [15] parsed documents as input to extract semantic relations and learn word sense embeddings to define triggers and arguments. Using compositional functions, event structure and argument representation are passed into a joint clustering framework. Mappings from FrameNet, VerbNet and PropBank allow for creation of trigger and argument cluster naming. Or in simpler terms, AMR documents are passed in and hierarchy based events are formed based on joint constraint clustering. To illustrate this concept, Figure 2.1 demonstrates a motivating sentence example with sample clusters used in the Liberal Event Extraction paper.

injure	Score	fight	Score	fire	Score
injures	0.602	fighting	0.792	fires	0.686
hurt	0.593	fights	0.762	aim	0.683
harm	0.592	battle	0.702	enemy	0.601
maim	0.571	fought	0.636	grenades	0.597
injuring	0.561	Fight	0.610	bombs	0.585
endanger	0.543	battles	0.590	blast	0.566
dislocate	0.529	Fighting	0.588	burning	0.562
kill	0.527	bout	0.570	smoke	0.558

Figure 2.1. Motivating example and sample clusters with similar types from Liberal Event Extraction [5].

The second tool is the Open Domain INformer (ODIN), created by Marco Valenzuela, Gustave Hahn-Powell and Mihai Surdeanu from the Computational Language Understanding (CLU) Lab at University of Arizona [4]. ODIN takes a recursive rule-based approach to NLP, compared to more traditional Machine Learning. The design philosophy of ODIN was to be able to capture simple dependency patterns and complex constructs, robustly handle syntactic errors, and execute fast [4]. Using the ODIN framework, the CLU Lab team was able to extend the established rule grammar into a real-world domain, which is officially titled as Reach (REading and Assembling Contextual and Holistic mechanisms from text) [16]. Reach is a statistical and rule-based event extraction tool for papers in the biochemical

domain [16]. The core of Reach consists for compact grammars that are able to recognize cellular processes in the form of biological entities and events [16]. Using this framework, Reach is able to discover previously unknown cancer signaling pathways. To demonstrate the rule-based system, a sample rule described in the ODIN paper is shown in Figure 2.2. More information about the modeling syntax can be found in the separate long-form ODIN description article [17].

```
- name: ner
   label: Protein
    type: token
   pattern: |
      [entity="B-Protein"][entity="I-Protein"]*

- name: Phosphorylation_1
   label: [Phosphorylation, Event]
   pattern: |
      trigger = [lemma="phosphorylation"]
        theme:PhysicalEntity = prep_of(nn|conj|cc)*
        cause:PhysicalEntity? = prep_by(nn|conj|cc)*
```

Figure 2.2. An example rule from the ODIN paper written in YAML [4]. It describes the basic syntactic structure for a phosphorylation event with token patterns using a trigger, theme and cause (optional). The protein defined as a simple biochemical entity.

2.1.3 Software Rationale

Compared to all the other Information Extraction software currently available, there are several reasons as to why we chose Liberal Information Extraction and Reach. First, both tools do not require any training data. Traditional methods of information extraction utilize machine learning techniques to extract information based on training data. For domains with little data available for training, the modeling approach offered by Liberal Event Extraction and ODIN offer an accurate alternative option.

Secondly, both tools are domain-independent. This means that the basic functionality of each tool is extensible to any possible field of knowledge. For ODIN, it offers individuals to build their own rule-based framework to extract event and entity concepts. Reach is simply one example of ODIN being applied to a complex, real-world domain [4]. For Liberal Event Extraction, one can utilize different AMR datasets to extract domain specific knowledge,

based on the reliability of the specified model. Additionally, the trigger list can be adapted to include new frame data based on the AMR data.

Last of all, both tools offer mapping capabilities to linked data. This is a common trend of a majority of information extraction tools. However, Liberal Event Extraction and ODIN make it incredibly simplistic to map to ontological terms based on the defined rule or frame. Reach demonstrates this capability by mapping its event concepts to biology based ontologies. Liberal Event Extraction events are formally grounded in frame-sets that can describe the context of words. The concepts defined in both tools provide explicit information related to its implied meaning in the context of a sentence, which is extremely important for mapping to explicitly defined concepts in ontologies.

2.2 Linked Data

2.2.1 Overview

The World Wide Web web was designed so that anyone could publish, share, and access information from non-centralized locations. Advances towards Web 2.0 have allowed information to be more usable and collaborative. Over the last decade, we have been slowly moving forwards to a fully connected, machine and human understandable Internet known as Web 3.0. This ideology was originally formulated by Sir Tim Berners-Lee, to allow information to be preserved and used more adequately. In order to realize this vision, an increasing number of applications have incorporated semantics to make the web smarter [18].

When you create or manage data, how does one signify the meaning of this data? Even though a dataset may exist, it needs to retain significance many years down the line. For example, if we had a database of electronically published biomedical documents, how do we derive basic information from this data? Every individual may have a different definition for what a document is, where it came from and the contributions established from this document may be unavailable without reading the document. Semantic data attempts to remedy this issue by integrating a standard method of common meaning, retrieval, and logical representation in the form of the Resource Description Framework (RDF). In other words, the semantic web helps data not to be so dumb [18].

A logical collection of data within the Semantic Web is known as an ontology. An ontology is utilized to describe information in a hierarchical manner and provide a common definition to the terms within. A knowledge graph is a compilation of information pertaining

to some task. Data within Knowledge graphs typically utilize terms from a multitude of different ontologies, providing further context to already common representations. All this information is stored in the previous format know as RDF, which takes form of triples (subject, predicate, object). An example of an RDF triple is demonstrated below.

dbpedia: Rensselaer_Polytechnic_Institute **rdf:**type **schema:**EducationalOrganization

We assume that new information may be discovered at anytime, better known as an Open World Assumption [18]. As more information is discovered, we can potentially link it to preexisting knowledge and create even more assumptions. To provide contextual information to our data, the semantic web utilizes a model known as the Provenance Data Model. Provenance basically allows individuals to better describe the data as a whole, which in turn provides assessment capabilities for the data [19].

2.2.2 Nanopublications

There are existing countless different methods for publishing of data on the Web. A Nanopublication is a community driven approach for semantically capturing a digital publication's assertion, provenance about the assertion and provenance describing the publication itself [7]. Each Nanopublication is uniquely identified as a Named Graph, which assigns a Uniform Resource Identifier (URI) to an RDF graph [7]. As described previously, a Nanopublication can be broken down into three separate components. The assertion graph contains the fact being claimed by the publication. For example, an assertion can identify that two proteins have some interaction with one another. The provenance graph provides context about the assertion graph. Provenance is able to details such as how the fact was generated, the time of creation, or anything that an individual may need to know about the assertion [19]. Finally, the publication information provides provenance about the Nanopublication itself.

We utilize Nanopublications to assert knowledge pertaining to some database or document. Additionally, Nanopublications provide a structure for further evaluation through each Nanopublication's source provenance. All this information is necessary if we desired to merge data from different sources into one location. Without proper provenance information, data loses its context and it becomes difficult to track the assertion derivation.

2.3 Knowledge Graph Construction and Refinement

As described previously, knowledge graphs help organize information systems into a more structured format. Knowledge graphs were theorized back in the advent of Artificial Intelligence and has practical usages in the World Wide Web to model Linked Open Data [20]. Knowledge graphs require some method of building information. Some common current knowledge graphs constructed from semi-structured data are DBpedia, Freebase, YAGO, and Google's Knowledge Graph [20]. Other methods such as NELL, PROSPERA, and KnowledgeVault utilize information extraction to convert unstructured or semi-structured data into knowledge graphs [20]. Overall, knowledge graphs play a critical role in the semantic web.

As described by Heiko Paulheim, it is important to consider the refinement necessary in knowledge graph construction [20]. Refinement typically concerns the correctness and completeness of a knowledge graph [20]. One of the more popular methods of refinement is Description Logic Reasoners. Description Logics provide a formalism for representing knowledge based on external and internal structure for inferencing between classes and individuals [21]. Some tasks in reasoning include subsumptions, which determines issues in subset denotations, and satisfiability, which checks concept expressions for empty concept denotation [21]. Description Logics are a widely studied field with numerous implementations to express correctness and consistency of graphs.

Another relevant form of knowledge graph refinement is inferencing. Inferencing consists of deriving new assertions based on a base set of assertions [22]. One common approach is inconsistency detection utilizing disjointness axioms for domain specific entities and events. We discuss implementing disjointness into the biomedical domain in our earlier work, defining possible types of inconsistencies for Reach output [23]. In addition to inferencing for evaluation, it can also refer to the generation of new knowledge from existing triples. For example, say we have triples containing HTML pages and information about that page. One can derive plain text from that HTML content and create new knowledge.

Knowing all this about knowledge graphs, why are they necessary for Natural Language Processing, as opposed to more traditional methods such as databases? Over the last decade, the emergence of deep learning and convolutional neural networks has shifted machine learning into utilizing more raw forms of data. Xander Wilcke argues that knowledge graphs can better express raw heterogeneous knowledge, such as entities and their relations, in com-

parison to feature vectors used in data science [24]. Based on this assumption, knowledge graphs portray a viable method of representing information and relations for a multitude of representation use cases.

2.4 Knowledge Graph Platforms

A knowledge graph platform incorporates the semantics of a knowledge graph into a framework for discovery, modeling and interaction. They frequently take the form of analytic platforms for large organizations. Some popular open source options for enterprise knowledge graph platforms include Neo4J, GRAKN.AI, StarDog, Apache Jena and Blazegraph. All of these platforms some capabilities in terms of data unification, support modeling, querying capabilities and description logic reasoning. However, the mentioned platforms are unable to meet all capabilities and perform advanced knowledge inferencing.

For the previously specified reasons, we chose to incorporate a new nano-scale knowledge graph framework called Whyis. The purpose of Whyis is to support domain-aware management, curation, publication, and creation of data-driven knowledge graphs from a wide variety of sources [8]. Knowledge is stored as nanopublications to allow for assertion generation and applicable provenance and publication association. Each nanopublication has a Uniform Resource Identifier (URI) and utilizes standards such as RDF, OWL and SPARQL [8].

The web stack consists of three major components: browser, server and storage/query. Browser allows for searching, editing, and viewing of information. Custom views can be integrated project specific interfaces. The server is built on top of flask and is where inference agents can implemented for new knowledge creation. Storage and querying capabilities are handled by celery, redis, blazegraph and file archiving of nanopublications.

The importance of this platform is it's extensibility and customization capabilities. In terms of data creation and management, Whyis offers many options. Go to a page and populate that with new knowledge, upload nanopublications or data files to associate with that page. Each page has an interface to interact with its facts, descriptions or nanopublications. Data can be loaded separately through the terminal and offers several data types. One can modify a page's view to change the information displayed and offer additional management capabilities to a user. With inferencing, existing data can be used to generate new knowledge. We utilize Whyis in this thesis to incorporate information extraction output and

demonstrate the innovative capabilities within the platform.

Chapter 3
RELATED WORKS

3.1 Information Extraction

Related to Lifu Huang's Liberal Event Extraction [5], he describes a method for event extraction without any manual annotations known as the Zero-Shot Transfer Learning framework [25]. Using event ontologies such as FrameNet, VerbNet, Propbank and OntoNotes, he is able to ground Abstract Meaning Representation to existing events. Additionally, Lifu has completed work in the field of Liberal Information Extraction for Entities as well [2]. It uses a similar approach with AMR annotations and hierarchical cluster embeddings to infer entities type from sentences.

We review the common approaches for event extraction from text [26]. Data-driven focus on words and n-grams. Popular supervised methods use machine learning techniques and run into several issues with large training sets being necessary. Inference systems for unsupervised documents focus on probability based distributions for prediction. Clustering is popular for unsupervised methods, allowing for prediction of new events. Knowledge-driven focuses on lexico-semantic patterns based on linguistic, lexicographic and human knowledge. They require less data for training the clustering phases, but require more development time due to domain semantics. Hybrid methods share similarities with data and knowledge approaches, but require a high level of expertise [26].

3.2 Knowledge Graph Construction

Several papers discuss the process of mapping data to knowledge graph, either using structured data from databases or unstructured data from documents. The ReDrugS (Repurposing Drugs with Semantics) project was designed to integrate from three biomedical based databases into a heterogeneous knowledge base and generate high-confidence drug repositioning candidates for melanoma treatment [27]. The assertions defined in ReDrugS are assigned probabilities based on quality of the methods used to create the assertion. This concept can be defined for a variety of domains to assess the reliability of assertions denoted. The base architecture of ReDrugS serves a similar function in comparison to the Whyis

platform and serves as an inspiration for this thesis work. We will learn more about the ReDrugS data flow in the following chapters.

The paper, "Knowledge Graph Identification", focuses on inferring of a consistent knowledge graph from an information extraction system, where extractors may have varying sources [28]. This is accomplished using probabilistic soft logic to perform reasoning over random values with soft truth values and first order logic rules. The underlying purpose is standardizing missing or incorrect Information Extraction output. It focuses primarily on entities and assumes ontological information about the terms are already available. Overall, it presents a different approach to the methods described in this thesis, while sharing some concepts like probabilistic logic from ReDrugS.

A related paper, "Constructing Knowledge Graphs of Depression", focuses on a similar topic, but uses the biomedical domain and a different methodology. The purpose of the DepressionKG system use multiple large knowledge sources to construct a smaller knowledge graph for clinical queries of a specific disease (depression) [29]. It is similar in concept to ReDrugS, where both have a graphical interface to interface with the constructed knowledge graph (query) and share rule-based inferencing. However, it is limited to the knowledge connections and semantic annotations defined by its data sources and lacks flexibility.

In terms biomedical knowledge construction specifically for entity linking, there are two relevant papers. The first paper focuses on an unsupervised grounded approach to entity linking for unstructured biomedical literature to 300 ontologies [10]. This process requires eliminates the need for large training sets, which is necessary for a domain like biomedical due to lack of availability. Entities in this study are assigned a higher probability based on edge weighting and non-collective entropy ranking. The second paper constructs patient-drug-disease graph linking entities in MIMIC-III to a Linked Data Cloud [30]. It utilizes similar biomedical knowledge graphs like all the previous papers and forms relationships between medical entities in MIMIC-III.

3.3 Workflow

The papers in this section all describe methods of building knowledge graphs from Information Extraction tools, the process involved with such task, and evaluation criteria of linked data. The first paper describes an automated method named HDSKG to discover domain specific concepts and relation triples from webpages [31]. This is accomplished us-

ing a rule-based dependency parser (NLP relation triple chunker) and a machine learning algorithm to estimate each triples domain relevance to create a domain specific knowledge graph. The interesting part of this approach is that they incorporate a knowledge repository into their evaluation process, independent of the extraction process. It also discovers that the created knowledge has potential implications on the already existing knowledge representation.

The second paper provides an incredible description of the basic template for an information extraction system workflow taking unstructured documents as input and outputting RDF [32]. The purpose of such a system should be to reduce the amount of time needed to read and comprehend domain specific articles. It suggests a flow of work in the following sequence: select journal, select area, select document number, read unstructured data to term list, create/generate RDF, and obtain results via SPARQL query. Additionally, it describes standard information extraction measures: Precision, Recall, F-Measure and Accuracy.

Chapter 4
WORKFLOW IMPLEMENTATION

4.1 Top-Level Data Flow

As described throughout this thesis, the purpose is to utilize varying Information Extraction methods to derive knowledge from domain specific documents. This process is visualized in a condensed data flow in Figure 4.1. The following sections in this chapter provide in-depth details about each step in this data flow and the process required to connect the steps.

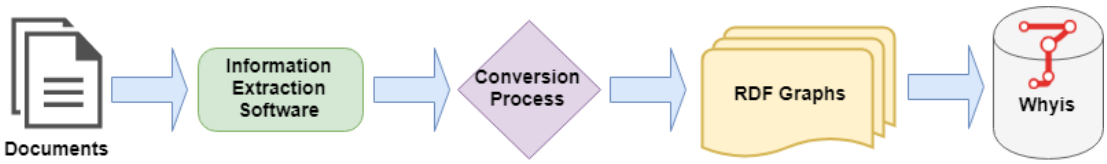

Figure 4.1. Top-level overview of data flow.

We start off with a domain-specific document corpus. Knowledge can extracted from the document corpus using different Information Extraction software. Next, we convert the Information Extraction output into a common format and apply a knowledge graph evaluation ontology to process information into nanopublications. These nanopublications are incorporated into Whyis, the nano-scale knowledge graph framework, which allows for curation, management and evaluation of our knowledge.

4.2 Document Collection

Before we can extract any knowledge, we need to choose and obtain a dataset. For this study pertaining to the biomedical domain, we chose to obtain our data from PubMed and PubMed Central. PubMed is a search engine and archive for millions of abstracts and citations from biomedical literature. PubMed Central (PMC) offers an archive of full-text journal articles, available electronically through the web. Both articles databases organize their articles by ID number, such as a PubMed ID (PMID) or a PubMed Central ID (PMCID). This is an important distinction to declare, because outside biomedical databases often refer to PubMed IDs when citing articles. However, not all articles in PubMed have an as-

sociated PubMed Central ID. Additionally, not all articles in PubMed Central are available via their API known as the E-utilities. Fortunately, PubMed Central provides a service that provides the full-text for articles via the E-utilities called the PMC Open Access Subset [33].

In order to collect our data, we start off with a list of PubMed IDs (PMIDs) pertaining to a dataset from an outside biomedical database known as iRefIndex (more details available about iRefIndex in Chapter 5, Section 5.1 and 5.2). We wrote a python script using the requests library (HTTP) to expedite this ID validation process. The script took a list of PubMed IDs as input and determined which PubMed IDs had an associated PubMed Central ID and had its full text available in the PMC Open Access Subset. The output of all PMCIDs with their corresponding PMIDs were provided to the user in a space-delimited text file. Based on this process, we included all PMIDs that met the appropriate criteria from the outside dataset for our official PMID data corpus.

We used the associated E-utilities command to download the specified PMCID list, which serves as the base for our data workflow. The format of the retrieved data was NXML, which is XML for the full text of the article, encoded in the Journal Article Tag Suite Document Type Definitions (NLM/JATS DTD) [33]. Having the data in an NXML format allows us to bypass the process of extracting text from PDF that has its own associated problems. In the next section we discuss how this data is used for each Information Extraction tool the extra steps required for Liberal Event Extraction.

For the purpose of clarification, the reasoning behind limiting our document corpus to those from the Open Access Subset is a result of the full text of documents only being available for these specific documents. If we were to limit ourselves to only PubMed documents and PubMed Central documents not in the Open Access Subset, the only text we could extract knowledge from is the abstract. Additionally, we limit our dataset to those PubMed IDs with corresponding assertions in the iRefIndex dataset. If we were to remove this restriction, we can apply our workflow to the entire PubMed Central Open Access Subset. In order to increase this dataset to documents outside of the Open Access Subset, we would require some alternative method of retrieving text from a digital file. Unfortunately, content and metadata detection is a completely separate issue that may require additional permissions for such content to be used in this manner.

4.3 Information Extraction

In this section we cover the process of converting our input data into the suitable format for each Information Extraction tool. Additionally, we go through the system model for each Information Extraction tool.

4.3.1 Reach

As mentioned previously, Reach is a hybrid statistical and rule-based extraction method [16]. Its focus is on recognizing cellular processes by recognizing biological entities and nested events. The architecture of Reach starts with preprocessing, by splitting the text into sentences, tokenizing words and completing part of speech tagging. Next, Reach searches for entities, simple events that operate on those entities and more complex nested events. Finally, Reach applies deterministic steps and a deterministic coreference-resolution system to enhance event output. We use the provided version of Reach on CLU Lab's Github Repository [34] and do not modify any portion of the code.

Reach takes a directory of NXML, CSV, TSV or TXT files as input. Reach is run using SBT, either as a server through command line or an internal web service via HTTP (with provided GUI). Reach offers several output options, which we will describe each briefly. The first option is FRIES, which is the standard option and outputs events, entities, contexts, passages and sentences. The second is Serial-JSON, where all document annotations and mentions are serialized (output is large). The third option is arizona (tsv/csv), which outputs only the events to a tabular format. For our workflow, we output results to FRIES format. The FRIES format offers a human and machine readable solution that provides context for the extracted knowledge, which is important for tracking provenance information [34].

4.3.2 Liberal Event Extraction

As defined previously, Liberal Event Extraction identifies events and event schemas using symbolic and distributional semantics [5]. The output of this system is able to represent event structures, event types, triggers, argument roles, and other relevant event information. The approach of the system can be summarized in Figure 4.2, which was used in the Liberal Event Extraction Paper. Liberal Event Extraction takes an alignment file and AMR documents as input into the system. This input is used to identify candidate triggers and arguments and construct the event structure semantics based on joint constraint clustering.

Joint trigger and argument clusters provide naming centroids to the events and outputs results.

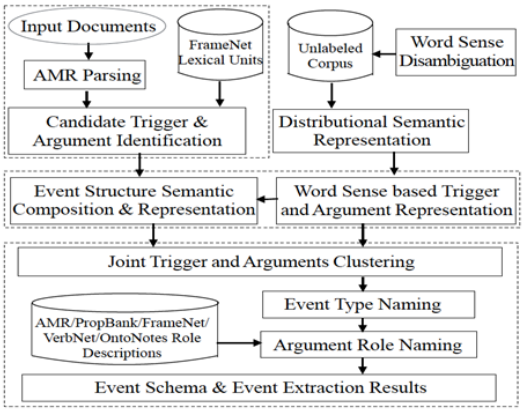

Figure 4.2. Overall Framework of Liberal Event Extraction, Figure 2 from original source [5].

In comparison to Reach, modifications need to be handled manually to create the Liberal Event Extraction Input. Liberal Event Extraction requires two files as input: an Abstract Meaning Representation (AMR) formatted document and an alignment file based off the AMR. AMR documents are sentences with a semantic hierarchy based representation. AMR Bank offers a corpora of human annotated documents available for download, including a corpus for cancer-related PubMed articles called the Bio AMR corpus. Unfortunately, if we want to convert our specific PubMed articles to AMR, we need generate AMR and the associated alignment on our own. To accomplish this, we utilize two parsers: JAMR and CAMR. JAMR is a tool able to perform semantic parsing, generation and alignment for AMR [35]. This tool allows us to generate AMR, without the need to manually annotate our date. The alignments are a necessary step to convert the sentences to AMR. This alignment serves a mapping from a dependency node to AMR concept in a sentence and are represented as a sequence of numbers. CAMR is a transition-based approach for AMR parsing using a tree-to-graph method [36]. Both parsers are able to produce AMR, but require the usage of JAMR for alignment generation. Parsers can trained on different datasets to be tailored to a specific set of data or use the default pre-trained model.

Although we describe the JAMR and CAMR parsers throughout this thesis as necessary components of the Liberal Event Extraction workflow, this is not the case. Any AMR parser that is able to convert sentences to this AMR format with associated alignments can be used

in place of JAMR and CAMR. In the case of our workflow where we use JAMR to produce the alignment, a separate AMR generator can be used in tandem with an alignment AMR generator, but is not necessary.

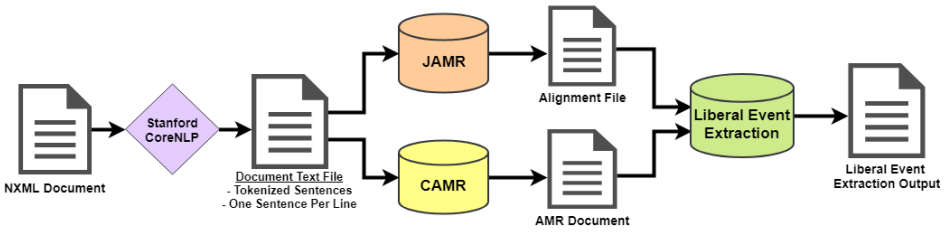

Figure 4.3. NXML conversion process into Abstract Meaning Representation and Alignment format, for usage as Liberal Event Extraction input files.

A figure of this process is demonstrated in Figure 4.3. We start off by converting our NXML documents into an AMR parser readable format, where every sentence is on its own line. This process is known as sentence tokenization and is accomplished in this workflow using the Stanford CoreNLP toolkit. Scripts were written in Java and python2 to extract these sentences from the NXML documents. The result of this entire process is NXML conversion to tokenized sentences. Once our sentences are tokenized, we can obtain our AMR and alignment files from the CAMR and JAMR parsers. Finally, we can use the AMR and alignment files in the Liberal Event Extraction tool.

4.4 Information Extraction Output Standardization

At this point in the workflow, each information extraction tool has created knowledge based on the input documents and we want to create nanopublications based on this information. The output is able to represent the events in a specified format pertaining to that tool. However, this information is represented differently for each information extraction tool. As opposed to implementing this process separately for each tool, we split this objective into two separate tasks. The first task is covered in this section, where we attempt to convert the output from each tool into a standardized event-based format. The second portion is nanopublication creation sequence, which will be covered in the same section.

An event has a generalized format that all events should contain: event type, argument, argument role, trigger and provenance information related to the event. Based on this knowledge, a standard format can be defined to represent all output from the different

information extraction tools. We utilize the FRIES output type defined in the Reach project as this standard format and add additional key-value pairs if necessary for other information extraction output [34]. The FRIES format defines itself as a light-weight, flexible and explicit JSON-based representation for textual objects and annotations [34]. The format is defined formally in its representation specification. The important part about this format is that can be adjusted to support other event extraction tools such as Liberal Event Extraction. The modified schema that we use for our workflow is outlined in Figure 4.4. Sample output produced for each information extraction workflow is provided in the Appendix B.

Although the schema more closely represents Reach output, modifications were made to support the additional information provided by Liberal Event Extraction. For example, the event's trigger and argument text provide mappings of their OntoNotes sense to the lexical databases of PropBank, VerbNet and FrameNet [5]. AMR incorporates core roles, which are accounted for in the Liberal Event Extraction output. We support the management of these terms through special keys in the general event mention schema. Some additional information provided in the Reach output is not available in the Liberal Event Extraction output, such as provenance information and the ability to reference external files (ontologies) for its entities and contexts. This supplementary information is an important distinction between the two workflows, as the external references are necessary for for later stages of the workflow evaluation.

4.5 Nanopublication Creation

This final portion of the information extraction output conversion process consists of extracting data from our JSON files and transforming it into semantically enabled RDF graphs, otherwise known as nanopublications. To accomplish this we utilize the base framework of the Knowledge Graph Evaluation System (KGES) [23]. The purpose of this project is to detect inconsistencies in large-scale heterogeneous knowledge graphs. However, we implement this software into our workflow for its base functionality of converting FRIES based input from Reach into nanopublications. KGES is able to transform all entity, event, passage, sentence and context output from Reach, but we will only be focusing on the event portion. We have also started the process of modifying KGES to convert Liberal Event Extraction output, due to differences in schema representation in Figure 4.4 and new event/argument types.

```
{
  "frame-id": unique identifier for event frame,
  "frame-type": "event-mention",
  "text": reduced sentence containing only aspects of the event,
  "type": event primary type,
  "subtype": optional event secondary type,
  "trigger": main word that expresses the event (should be in text),
  "trigger-role": trigger frame role mapping (Liberal)
  // arguments: textual argument list for this event (average of 1-3),
  "arguments": [{"object-type": "argument",
                 "text": word or phrase of the argument mention,
                 "text-role": text frame role mapping (Liberal)
                 "argument-type": object type of the argument,
                 "type": syntactic role of this argument,
                 "argument-relation": role in AMR format (Liberal)
                 "index": argument number (to convey ordering),
                 "arg": pointer to frame-id describing argument}, ...],
  "index": sentence-local number for mention (optional),
  "start-pos": Text start position of mention in sentence,
  "end-pos": Text end position of mention in sentence,
  "sentence": reference to sentence frame-id,
  "verbose-text": full sentence string that contains the event,
  "is-direct": represent negated information (Boolean type),
  "object-type": "frame",
  "object-meta": {"component": Information Extraction tool name,
                  "component-type": "machine",
                  "doc-id": PubMed ID or PubMed Central ID,
                  "processing-start": program execution start time,
                  "processing-end": program execution end time,
                  "object-type": "meta-info",
                  "organization": institution of creator}
}
```

Figure 4.4. Event mention standard representation formatting.

All assertions contain an event type, event mention, original sentence, and trigger. The event and argument types are grounded to ontology terms to provide standard representation. The ontologies used for the event and argument types are MI, GO, SIO, and NCIT. The formal list of event and entity type groundings for the biomedical domain are provided in Appendix A. Depending on the event type, there may be a specific number of arguments and each argument is grounded to SIO relations. Interactions have a target and participant, while complexes have component part(s). For Reach data, entities are already recognized and

correspond with biomedical based ontologies. Some of these ontologies include Uniprot, Gene Ontology (GO), ChEBI and Uberon. A full list of can be located on CLU lab's bioresources repository [37]. Provenance and publication information are grounded to the KGES ontology and the PROV ontology. The provenance information describes how the event was extracted in the information extraction tool. Publication information describes which tool was used, the generation time, the PubMed source and its location in the document. Although some of these terms were defined in previous work for KGES, we were required to expand the term list for undefined mention types in Reach and for all the new mention types from Liberal Event Extraction.

The assertion of each nanopublication we create is the fact that each event is declaring. For each assertion, they have an event type, the source sentence, a delineated event phrase, a trigger word, the arguments and each argument's role. Provenance information describes information such as the time of the assertion's generation, how it was found by the information extraction software, the event's start and end position in the sentence, and possible event directness, and the frame type. Finally, publication information describes the nanopublication's metadata: the information extraction software used to generate the events, the creator of the information extraction software, and the PubMed document ID. More information about these terms can be found in the KGES ontology.

One concern with this grounding process is that all these event and entity types needed to be determined manually. For example, if we found an entity mentioned in an argument of type protein, it was necessary to look through the relevant domain ontologies find an external reference that best fit with the type we had. This process needed to be repeating for any new term we found. This process was simpler for the Reach output because the mention types are specified in its documentation and is only relevant for the small number of implemented rules. However, Liberal Event Extraction is grounded in the broader framesets of AMR, FrameNet, VerbNet and Propbank. This meant that the number of mention types that needed to be annotated was much larger and we chose to only ground a subset of these terms as a result. We address this problem in Chapter 6 as part of the workflow enhancement analysis section.

4.6 Knowledge Construction and Inferencing

The final portion of our workflow utilizes the Whyis knowledge graph platform. We have created nanopublications of our information extraction output that can be merged together for large-scale data management and inferencing of new knowledge. Using the created nanopublications, we can load our knowledge into Whyis using its basic load command or to specific pages in the Whyis user interface. For our workflow, we use Whyis for its knowledge publishing and management for evaluation purposes. The process of evaluating our results is described in the following chapter. Future work and discussion will discuss the current inference agents in development for Whyis. In order to use Whyis, instructions are included on its Github Repository to install and configure a personal environment.

The formal definition of inference agent is a process integrated into Whyis to provide additional knowledge or perform some task on pre-existing knowledge [8]. An inference agent is called upon update of the knowledge graph in Whyis or based on a timed schedule. Some example tasks for inference agents include tokenization of sentences with the NLTK package, XML parsing and triple creation.

Description logic reasoning is an essential field of study that provides potential influence in the task of knowledge inferencing. In the case of our workflow, we define numerous assertions that may initially seem unrelated without prior context. These assertions may have potential connections with other assertions or may be explicitly defined to be disjoint with one another. We can establish these connections using the commonly defined task of a description logic reasoner. Some of these tasks include: instance subsumption closure, property inversion closure and class equivalence closure. Description logic reasoners are often grounded in languages that can describe classes and the relation between classes. One common example is OWL DL, which is a common language for encoding the aforementioned class relations. By loading ontologies encoded in OWL DL notation, we can provide context to our previously created knowledge from information extraction and discover the hidden relations.

4.7 Connected System Design Data Flow

This chapter has described the process of extracting knowledge from a PubMed document using information extraction software and conversion of that knowledge into a semantically enabled format for further curation. In figure 4.5, we visualize this process for a

fine-grained perspective based on the information provided throughout the chapter.

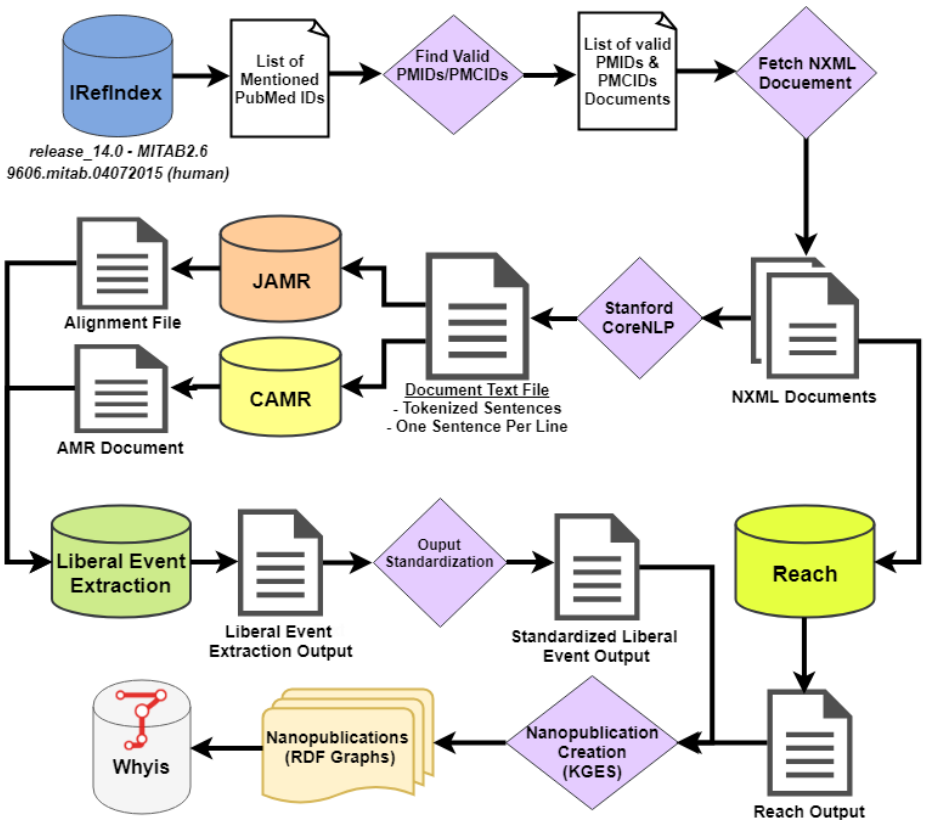

Figure 4.5. Full data workflow diagram. Includes data collection, information extraction processing, output conversion and loading into Whyis.

Chapter 5
EVALUATION

5.1 Approach Overview

Previous studies for event extraction often have domain experts review the accuracy and usefulness of events to measure the performance of output. For this study, we forgo the manual method of evaluation, in favor of a fact based approach. Numerous databases exist that provide data pertaining to domain-specific output. We utilize a protein interaction database called iRefIndex [38] to provide facts related to our expected RDF graph output and measure its accuracy. To elaborate, the workflow is able to extract interactions between objects from pubmed documents. Interactions from the biomedical domain consist of cells, proteins, genes and more. The iRefIndex database provides annotations about protein interactions and specifies which pubmed documents pertain to that annotation. We can determine the accuracy of our system by determining how many events from our output have matching annotations in the iRefIndex database.

In contrast with the evaluation metrics used for domain specific information extraction, we perform an automated factual evaluation approach. This is important for large-heterogeneous knowledge graphs, as manual annotations become more labor intensive as output size increases. Other common methods of information extraction include machine learning to identify common parts of an event's structure and/or to construct semantic representations [16]. Although this type of evaluation is necessary for determining the precision of an information extraction system, we take an alternate approach inspired conceptually by to those used in the ReDrugS project. The events found from an information extraction tool can be compared with those in a factual database to determine the similarity with already validated knowledge. From previous studies on a particular information extraction tool, we can expect some precision value to be defined. From a factual database, we can estimate each fact's legitimacy defined by information in its provenance. In summary, we perform evaluation by checking for expected knowledge and by incorporating probabilistic factors into the analysis.

5.2 Assertion Data Conversion Process

For this study, we chose to analyze the Homo Sapiens (human) taxonomy dataset (taxonomy id 9606) from iRefIndex [38]. As mentioned in Section 4.2, we are able to extract a list of PubMed ids from this dataset. Unfortunately, must validate each PubMed ID to confirm that each has an associated PubMed Central ID and is also available in the Open Access Subset. As a result, we identify 3115 PubMed IDs that meet the appropriate criteria. For this thesis, we only evaluate a small subset of this original data corpus, specifically 105 PubMed IDs.

The data from iRefIndex is stored in tables available via its website. Before we can compare this information, we need to convert the iRefIndex data into a nanopublication format, similar to the data from the event extraction workflow. This process is formally defined as semantic ETL, where data is transformed to a format defined based on a template file. This procedure of semantic ETL for the iRefIndex data has been previously implemented by James McCusker for the ReDrugS project. The original scripts are located on the Github repository for ReDrugS [27]. We created a modified version of its template to use the modern version of the nanopublication schema and to include some additional provenance information about the iRefIndex data. The data conversion process is visualized in Figure 5.1.

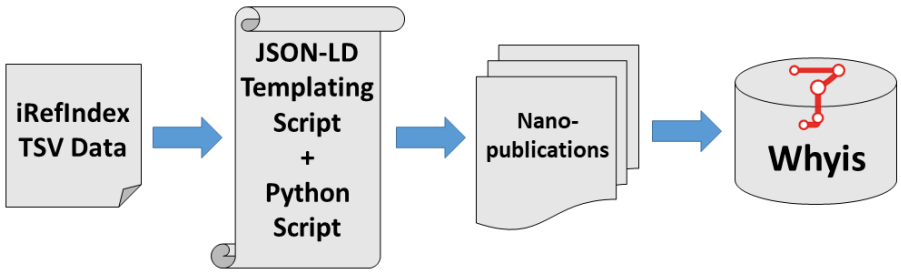

Figure 5.1. iRefIndex data conversion process. Files and diagram used are a modified version that were originally created for the ReDrugS project. This process is formally defined as Semantic ETL.

In order to evaluate the data from the information extraction workflow and iRefIndex workflow, we load both datasets into Whyis to distinguish matching nanopublications between the two datasets. A series of SPARQL queries are able to distinguish matching RDF graphs and return all specified triples. SPARQL is a standard semantic RDF query language that is able to retrieve and update RDF graphs loaded into an HTTP service endpoint [39].

Using SPARQL queries, we are able to obtain triples related to our evaluation criteria.

5.3 Evaluation Criteria

We based our evaluation on the simple criteria that an information extraction should be able to produce the same knowledge as a factual database, provided the same document as input. With that in consideration, specific details need to be covered in regards to the evaluation for our datasets. For this evaluation, only Reach is considered for qualitative assessment. Reasoning for not including Liberal Event Extraction is provided in the following chapter, results and discussion. Additionally, we limit the data from our factual assertion database, iRefIndex, to the human taxonomy dataset. The purpose of only focusing on the human taxonomy was to limit the scope of our study and to better support the rules created for Reach, which apply primarily to extraction of cancer signaling pathways.

A fact from iRefIndex is considered to have a match if there is an event from Reach output that shares a similar event type, equivalent arguments, and has the same PubMed document source. For simplicity, we limit our evaluation to events of type interaction/association, a connection between molecules. iRefIndex data consist of approximately 70% interactions and 30% complexes. We do not include complexes into our evaluation due to schema differences between the iRefIndex data and the Reach data. The interactions within the iRefIndex often have multiple assertions pertaining to the same interaction between the same molecules. The differentiator is the experimental method it was derived from and a specific interaction type. Due to Reach's current inability to determine an interaction's experimental method and grounding differences between the interaction types, we choose to limit the iRefIndex data to unique assertions.

A unique assertion is defined by the target and participant in the interaction, considering the experimental method and specific interaction type to be non-essential for basic evaluation criteria. All event types have a superclass of type interaction, so we chose to declare a matching regardless of more specific event type. The reasoning behind this decision was to account for grounding to multiple ontologies for event types and to keep our evaluation fairly broad. However, it is necessary to realize the value lost by excluding these more specific event types from evaluation. The argument roles are grounded to the same SIO terms of target and participant, so our evaluation does account for the role of each entity in the interaction.

We propose four evaluation measures: explicit accuracy, relaxed accuracy, precision and recall. Explicit accuracy requires that both events are of type molecular interaction and share the exactly the same label and linked data reference. Relaxed accuracy follows a similar procedure as explicit accuracy, but accounts for syntactic differences between the arguments. Some metrics that qualify for inclusion in relaxed accuracy include: contrasting linked data reference with the correct labellings, contrasting labels with the matching linked data references and incorrect target/participation ordering. Precision expresses the percentage of our matches that were identified incorrectly. Recall is used to determine which assertions from the iRefIndex dataset had an associated interaction from the Reach output, but was not identified as such. In other words, the recall statistic helps delimit false negatives. To gather these statistics, we have written a series of SPARQL queries to capture such characteristics.

It is necessary to realize that this evaluation is not exclusively intended to test the accuracy of each information extraction software and the overall workflow. Accuracy based on correlation to ground truth is limited in scale regarding applicability to realistic applications. By comparing the results of factual assertions and automatically derived knowledge, we can distinguish what matches were discovered and how our workflow can be improved.

Chapter 6
RESULTS AND DISCUSSION

6.1 Analysis

In this chapter, we discuss the results of our workflow implementation for the Reach and Liberal Event Extraction methodologies. Using Reach, we were able to successfully extract events from a subset of PubMed documents referenced in the iRefIndex database for the human specific taxonomy dataset. Inclusion of nanopublication formatted iRefIndex and Reach output into Whyis provided a procedure to evaluate our findings. Overall, this evaluation provides a pathway for further development of our workflow to improve and expand its potential usages. Later in this chapter, we provide insight into the various implications of our evaluation results and the workflow implementation itself.

We simplify our evaluation to the criteria established in Chapter 5.3. Using the data collection validation portion of the workflow, we limited the original iRefIndex human taxonomy dataset with 673,100 assertions and 41,971 unique PubMed IDs to a publically available subset with 93,315 assertions and 3,115 unique PubMed IDs. For our evaluation subset of 105 PubMed IDs (originally from the 3,115 public set), we found 1,091 nanopublications describing assertions of type interaction or association. We limit these assertions to those with a unique target and participant interaction/association. The evaluation is then computed by querying our datasets and computing the accuracy, precision and recall metrics. The results of this approach are showcased in Table 6.1.

Table 6.1. Simplistic results for matching interaction events between iRefIndex and Reach for PubMed document subset.

Approach	Unique Assertions
Total Interactions	857
Explicit Matches	46
Explicit Accuracy	5.37%
Filtered Relaxed Matches	138
Filtered Relaxed Accuracy	16.10%
Total Relaxed Matches	209
Total Relaxed Accuracy	24.39%

Two measures that are left out of Table 6.1 are precision and recall. For our precision metric, we consider if any of our interaction matches were identified incorrectly. Based on the specificity established in our SPARQL query, our precision value is considered to be very high (around 99.99%). However, this is dependent on assuming the assertions created in iRefIndex are correct. The experimental method and source database can also impact the reliability of each assertion. As for recall, this statistic is dependent on false negatives. We formally define a false negative as any assertion in the iRefIndex dataset that had a corresponding assertion from the Reach dataset, but was not returned in the evaluation query. Unfortunately, a time-intensive manual process is required to distinguish the difference between a true negative and a false negative assertion. This process consists of using an iRefIndex assertion, determining if an assertion in the Reach dataset is similar to the iRefIndex assertion, and to what extent are those two assertions similar. If we consider the precision to be very high, we can also assume that recall is similar to accuracy and would increase as more unidentified matches were found.

The accuracy values may seem low at a glance, but manual observations about result differences provide sufficient justification for these values. Several false negative interactions are illustrated in Table 6.2. One major issue we discovered was related to co-referencing, where two objects are labeled or represented differently, yet share the same meaning. This is a common problem within entity linking for biomedical literature due to the immense amount of terminologies and abbreviations. [10]. Although it may be easy to ground all terms to the same terms for one information extraction workflow, iRefIndex may use a completely different set of groundings for its entities, causing accuracy of our evaluation to drop considerably. Another issue is missing information from both sources. Reach was not able to find external references for its entities and grounded it to a default name-space of UAZ. iRefIndex does not have aliases (labels) for all of its unique identifiers, relying only on its external reference to find a matching assertion. Overall, the large number of potential unidentified matches relies on specificity of the found interactions and a lack of related knowledge about each interaction.

The purpose of the relaxed matching was to capture two interactions that were originally supposed to produce a match and avoid the previously mentioned issues. We implemented a few solutions via SPARQL filters to capture specific match types such as the incorrect target/participant order, non-matching external references with the same labeling,

Table 6.2. Manual observations examples about misidentified matches (False Negatives) for interaction/association events between iRefIndex and Reach for PubMed document subset.

PubMed ID	Reach Object		iRefIndex Object	
	Target	Participant	Target	Participant
9874563	VPR pfam:PF00522	TFIIB uniprot:Q00403	GTF2B uniprot:Q00403	VPR refseq:NP_057852
12193273	PDE4D5 uaz:UAZ00064	RACK1 uniprot:P63244	PDE4D uniprot:Q08499-6	GNB2L1 uniprot:E9KL35
15328530	EEA1 uniprot:Q15075	Rabenosyn-5 uniprot:Q9H1K0	RAB5A uniprot:A0A024R2K1	ANKFY1 uniprot:Q9P2R3
15522123	IRS-1 uniprot:P35568	HDAC2 uniprot:Q92769	IRS1 uniprot:A0A024R499	HDAC2 uniprot:Q92769
15796781	IPORIN uniprot:Q8N2Y8	HDAC2 uniprot:P62820	RUSC2 refseq:NP_001129471	RAB1B uniprot:Q9H0U4

and special external reference extensions. Unfortunately, the time process of creating these rules is time and labor intensive. This provides further rationale for description logics built into Whyis inference agents. To demonstrate this point, let's take the false negative example from PubMed ID *12193273*. Here we see both sides have a version *PDE4D* as their target. However, Reach couldn't find an external data representation for *PDE4D5*. The uniprot page for uniprot:Q08499 indicates *PDE4D5* is an isoform of *PDE4D*, which could have been detected with inclusion of the uniprot page. As for the target portion, we manually discovered that *uniprot:E9KL35* shares a 100% similarity to *uniprot:P63244*. To provide further clarity to the uniprot similarity statistic, the uniprot knowledge base organizes its entries into clusters based on quality of the entry, annotation score, organism and length of the sequence. This ensures that all sequences in uniprot are covered into a common resolution space that attempts to reduce redundancy [40]. By incorporating these clusters into Whyis, Description logic reasoners can be configured to recognize these similarity scores and assign OWL equivalences as deemed appropriate. This similarity process requires completely different criteria when computing similarity between two different ontology sources (most likely automatic similarity calculation). Overall, these are just some of many examples that showcases the potential of Description Logic Reasoning and knowledge inferencing incorporation into Whyis and the resulting possible increased accuracy.

The process of analyzing our results were primarily through manual observations, as

Table 6.3. Error categories for evaluation results based on manual observations.

Error Category	***Description***
Coreference Misidentification	Assertions with objects labeled or represented differently, yet share the same meaning (broad concept)
Entity Reference Relevance	Two entities sharing the same argument role, but are grounded to different external references (either to same or different ontologies)
Entity Alias Relevance	Two entities sharing the same argument role and external reference, but each entity has a different alias (labeling)
Association Equivalence	Manual assertions that can be derived from multiple IE assertions (Example: $X \rightarrow Y$, $Y \rightarrow Z$)
Unknown Reference	External reference grounding error for arguments. Common example include unknown ontology reference or no associated alias.
Unreadable Source	Manual assertion derived from a source not provided to the IE software. Common examples include diagrams and charts.

described previously. This problem arises from the specificity of our SPARQL queries, which can only determine results based on the data it had already and has no innate ability to form missing connections. From the manual observations established in this section, we provide categories for probable sources of error in Table 6.3. Beyond those errors described in our evaluation process, there are several other probable errors that are propagated throughout our workflow. The first error is schema differences between the information extraction output data and the iRefIndex data. This can be caused by usage of different ontology groundings, exactness of the mention types, and the unavoidable issue of the argument relations being defined differently in our datasets (example: "has participant" and "has agent"). This schema problem extends beyond the scope of this thesis and has potential solutions through. The second error pertains to the original data source and the information available for that analysis. As described in our error categorization table, an information extraction tool is limited by the text it is provided and its inability to comprehend diagrams. In the case of Liberal Event Extraction, events are stored in sentences and coreferencing resolution is required to connect an entity mentioned in multiple sentences. This problem extends to various facets of the natural language processing domain such as the inference level (paragraph, sentence, etc.) and semantic interpretation of diagrams. The third and final error is the accuracy of our information extraction software. Without comparing our results to manual assertions,

we are dependent on the information extraction software itself to provide correct events. An incorrect event without any associated reliability scoring can cause serious harm if the event is misused for a real-world use case.

6.2 Implications

The generated results and general implementation process has prompted several interesting observations. Both information extraction workflows have several benefits and downsides in comparison to each other. Reach offers a rule-based approach specifically for the biomedical domain. It is incredible simplistic in its use, not requiring a computationally intensive process to compute results. The external references of its entities linked to ontologies simplify an otherwise costly transformation process to RDF graphs. However, Reach is only able to succeed only in respect to its defined events. Creation of new rules for other domains requires extensive knowledge of that domain, logistic knowledge of how distinguish that event and programming experience to actually implement the rules.

Although Liberal Event Extraction is a more general system in comparison to a rule-based system, Liberal Event Extraction has major benefits in comparison to other commonplace Information Extraction tools. The Liberal Event Extraction can be rapidly adapted to new domains, without the need to define new syntactic rules. Grounding results to meaning representation and semantic role descriptions in FrameNet, VerbNet and Propobank provide methods for standard term mapping and can discover rich event schemas not manually defined [5]. There is an abundance of potential in regards to new knowledge generation from the output of this system. We speculate that incorporation of entity linking to common outside representation sources and a deeper understanding of the frame semantics can lead to discovery of the previously mentioned rich event schemas for more specific knowledge generation tasks. Additionally, the wide variety of events captured by the system can be of major benefit to an inference based system, such as Whyis.

As discussed in the previous section, we found several inconsistencies in the results that lead to a more accurate depiction of the accuracy. One particular observation not mentioned is the occurrence of output errors and incomplete data. For example, Reach is not able to identify all entities and associates such entities to an undefined name-space. Some of the assertions generated via iRefIndex may come from details not provided in the main text of the PubMed document. Any assertions that are derived from a non-textual form like a

picture, chart, or additional material put information extraction tools at a disadvantage, as that knowledge may not even be available to the information extraction tool. Despite the fact that an assertion from Reach may not have a match in iRefIndex does not imply that the knowledge is incorrect. In actuality, information extraction data can be used alongside their source ontologies to find previously undiscovered connections. These discoveries may improve the accuracy of our results by filling gaps in prior created knowledge.

In consideration of the approach taken for the implemented workflow, there are more factors to examine that may further improve the results of our work. Databases such as iRefIndex have their own pre-established confidence metrics. Some of these metrics include the lowest number of distinct interactions from a referenced PMID, the highest number of distinct interactions from a referenced PMID, and the total number of unique PMIDs used to support the interaction. Incorporating such features into our information extraction knowledge would be immensely useful. This feature can be incorporated into Whyis as an inference agent that automatically generates these confidence metrics on a timed interval. On a similar note, more provenance information can be integrated into our nanopublications for both sources. For example, the interactor type is already available in the output data and can provide more context to our entities. Other information such as PubMed ID document provenance is necessary to provide context about our assertions. However, this would require some additional automated processing of its XML representation to obtain this data.

It was mentioned in the evaluation criteria that we chose to assume that any event type with a superclass of type interaction was considered for potential matches. This excluded evaluation of the more specific event types that were applied to the assertions. Further evaluation can be performed on those interactions with matching arguments and roles to determine how accurate these event types were to each other. Additional steps may be taken to discover the impact of reversed argument roles and the directness (negation) of the interaction itself.

The observations made about the output differences provide several significant next steps for our work. First of all, an inconsistency pass on our information extraction data is mandatory to ensure stability knowledge. Inconsistency checking also provides more evaluation potential and can lead to entirely separate study. Secondly, incorporation of inference agents to create new knowledge is necessary to fill missing gaps in representation and may lead to new interactions. Finally, using the framework established in this thesis, application

to a real-world use case is an essential validation step. Some possible directions for such a study may include knowledge summarization and justification, a QA system related to interactions pertaining to documents, and incorporation into an entirely new topic domain.

6.3 Implementation Difficulties

The information extraction tools were fairly straightforward to use without opening the metaphorical black box. However, each tool's integration into the workflow had numerous associated issues. Despite the fact that Reach was incredibly simple to use and generate results, it suffered from non-standardized output conventions. For example, when representing a complex in an event argument, it changes the key from 'arg' to 'args', yet the documentation does not formally define this difference. Another issue related to sentence text including newlines and other invalid characters. In general, resources for Reach were available on its project page, but were often organized in non-logical format and specific details tended to be scattered across multiple resource pages.

Liberal Event Extraction is highly dependent on an Abstract Meaning Representation (AMR) parser, of which there aren't many publicly available that don't rely on the alignments of the JAMR parser. These publicly available AMR parsers, CAMR and JAMR, had several issues in respect to its usage not being fully functional or documentation not being descriptive enough to debug issues. Domain-specific models tend to not be publicly available.

In addition to the AMR parser issues, we encountered several problems related to the Liberal Event Extraction software itself. First of all, Liberal Event Extraction is not optimized for machines low-end performance capabilities, so a separate machine with better performance capabilities would be necessary to generate results. Secondly, numerous output modifications are necessary to produce nanopublications. Each event type, trigger and argument is grounded to terminology defined in AMR, Liberal Event clusters, and the utilized frame-sets (FrameNet, VerbNet and PropBank). Explicit definitions of their source external reference would greatly improve provenance within its output. Additionally, Liberal Event Extraction has some difficulties describing the explicit word/phrase in the sentence for triggers and arguments. For example, the output may indicate an argument of type enzyme, but doesn't specify the word it corresponds to in its output. Without modifying the program, we could theoretically bypass this by comparing its position in the original AMR. Some other issues pertain to phrase and event mention depiction in the output. All of these issues

pertaining to the Liberal Event output puts the burden on our workflow to incorporate the missing information and provenance into the conversion process. Lastly, evaluation is often associated with separate portions of the Liberal Event Extraction workflow, which leads to difficultly in considering the impact of each step.

In summary, we encountered several issues regarding the Liberal Event Extraction workflow implementation that went beyond the thesis scope. As a result, we could not perform qualitative evaluation and fully implement the Liberal Event Extraction workflow. Various features included in the Reach software such as sentence tokenization, entity linking, and output standardization to a common data format had to be handled manually outside of the Liberal Event Extraction software. Although the Liberal Event Extraction workflow was not able realize full implementation, arrangements have already been constructed to remedy portions of the proposed workflow and incorporate the ideology for smaller scope identification.

Pertaining to both information extraction methodologies, it is important to consider the amount of time necessary to manually annotate an information extraction's output with external ontological references. This process requires a domain-expert with awareness about how the knowledge is represented in the information extraction output and must be knowledgeable about the appropriate ontologies for alignment. Possible future work can be accomplished to automate an otherwise time-intensive manual grounding process.

The Whyis knowledge graph framework is still a fairly new platform that began development in early 2017. Development of the platform and documentation is actively ongoing into 2018, which means the system is constantly changing and can be difficult to implement features. The groundwork of inference agents provides an initial path for implementation into Whyis. However, a lack of resources makes this a labor-intensive task. As the platform adds more features and becomes more popular, the implication of implementing portions of our workflow in Whyis becomes more beneficial for a potential user. Some of these features that would be pertinent to our study are the source ontology upload/view, knowledge graph editor, faceted browser and the inference description logic reasoner. To conclude, the topic and overall domain of this thesis work is incredibly broad with several potential paths going forward.

6.4 Workflow Assessment

Implementation of two different information extraction methodologies and evaluation of their linked data representation allows us to delve further into the various aspects of our approach. In summary, we were able to successfully extract knowledge from the biomedical documents and find connections to manually created assertions. However, we noticed several issues when comparing the two datasets that could have improved upon. What was the cause of these issues? How can our approach be augmented to remedy concerns in our analysis and allow for applicability to real-world use cases?

To start off, we need to discuss negative aspects of our approach. The entire workflow construction process is labor-intensive. In order fully implement such a system, each component from the workflow must be connected to eventually create a representation usable in a knowledge inferencing system such as Whyis. Information extraction software itself is often focused providing results in its own specific representation, further increasing the difficulty of creating a standard representation format. The event types created from information extraction required manual grounding to external references, which also depends on the context of how these events are derived. Manual assertional databases often have their own representation and grounding preferences. This further complicates circumstances when we attempt to compare the results of our workflow output. We mention previously that description logic rules and upload of source ontologies utilized for representation can remedy some concerns over unidentified assertion matches. However, data starts to accumulate quickly as we upload larger ontologies and more nanopulications, requiring supplementary modifications to the Whyis to avoid current Blazegraph limitations. Even with all this in mind, we still haven't addressed concerns related to provenance about our assertions. Some of these concerns include missing document metadata and experimental derivation for the biomedical domain. Extension of this workflow to future domains is possible, but requires new modeling requirements specific to each information extraction software and manual external reference grounding of newly identified event types currently.

Although there appears to be numerous issues with our approach, the developed workflow is imperative for implementation of a knowledge extraction/inference system and our results provide critical information for additional necessary steps to apply such a workflow to a real-world use case. The idea of knowledge extraction and conversion into a linked data format has been accomplished before. Incorporation of this knowledge into Whyis in nanop-

ublication format can lead to curation, management, and creation aspects that haven't been considered in previous studies.

6.5 Information Extraction Methodology Enhancement

Throughout this chapter, we have recognized several issues with the implemented workflow and our results. In order to improve our current workflow, we can provide potential solutions to these problems in the form of a revised workflow. To visualize these alterations, we provide a revised version of Figure 4.5 in Figure 6.1. There are three major adjustments included in this diagram: document provenance nanopublications, information extraction updates, and Whyis inference agents. Separate nanopublications about the original PubMed documents are necessary for source metadata management and potential use cases. Information extraction updates are specific to each software to improve output usability within the Whyis platform. Liberal Event Extraction requires references to the associated frame-sets and some provenance about the output itself. These changes for Liberal Event Extraction can be made within the software itself or separate post-processing of the output. Reach requires additional rules to be created within the software to distinguish the experimental methods mentioned in the PubMed document. The final adjustment is the inclusion of currently in development inference agents within the Whyis platform. Custom views provide a visual interface for project specific knowledge. The description logic reasoners are necessary for creation of inferred/expanded knowledge and inconsistency checking. Faceted browsers can display all the created knowledge in a searchable manner. The source ontology uploader is required to discover relationships similar to our already created knowledge. It is necessary to note that all these suggested adjustments related to Whyis inference agents have yet to be officially tested on for our proposed workflow. Successful data loading and development tests of these mentioned inference agents will lead to full incorporation within the following months.

Unfortunately, it is not entirely simple to adjust this workflow to different domains. The entire data collection and document provenance conversion are completely dependent on the dataset source. New rules need to be created to support different domains in ODIN (the base of Reach), as well as new manually created event type groundings. However, the framework we have built for the Liberal Event Extraction can be applied to any domain, regardless of the data source. Automated event type grounding is still necessary to remove

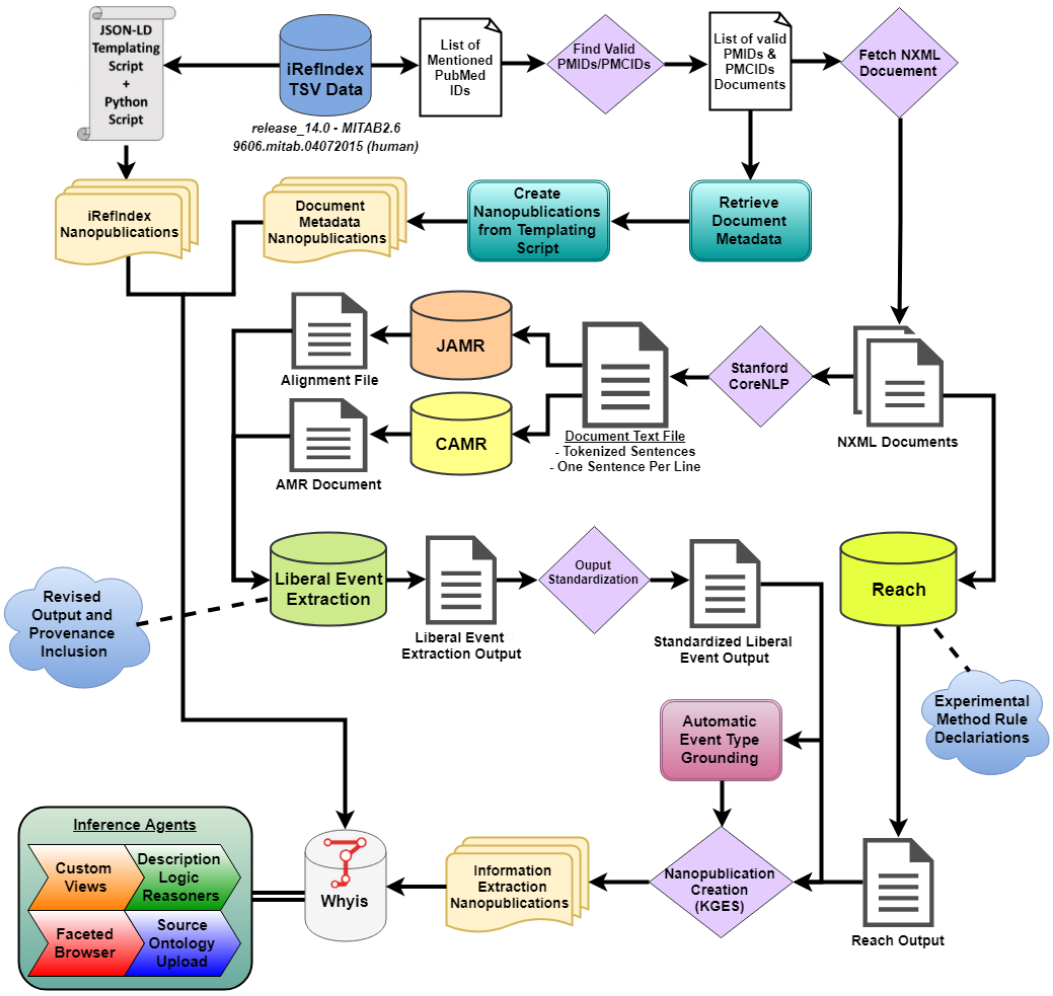

Figure 6.1. A complete version of the workflow implementation, including essential revisions based on observations made in evaluation analysis.

the manual grounding aspect of the workflow. To allow our workflow to support multiple domains, we would probably take an approach similar to that mentioned in the unstructured data to RDF paper [32]. This would allow the user to specify the area of study in which the documents associate to. Additionally, if we wanted to incorporate data not from PubMed Central, adjustments are necessary for the data collection portion. Assuming that the data is in a standard, digital format such as PDF, some text processing software must be used to extract the text and organize its content into a usable format for ODIN and Liberal Event Extraction.

As for automation of the entire workflow, the current workflow implementation handles all processing and creation of the data into nanopublications outside of Whyis. However, we believe that incorporating seprate portions of the workflow into Whyis is the best path

forward. The first major portion would be a PubMed validator and retriever. As an example, a user might enter a list of PubMed IDs into a custom Whyis view. The page would then return those PubMed Central IDs in the open access subset and retrieve the document's XML representation and metadata. Next, an inference agent can be triggered to convert the metadata to nanopublications and create nodes (pages) for each PubMed document. Another inference agent could be triggered to take the XML PubMed document and run it through each information extraction software. Unfortunately, each information extraction software is computationally intensive and the Liberal Event Extraction workflow requires outside pre/post-processing. Therefore, the two best options would be utilizing APIs to handle this computation on another machine or to provide the user with the retrieved XML and documentation to run this portion of the workflow on their own machine (outside of Whyis). This multiple option approach can be applied to the standardized information extraction output into nanopublications. Further optimizations are currently being made to the Whyis nanopublication load process, so implementation of data loading is dependent on the most recent version of Whyis. The rest of the work is dependent on implementation of the faceted browser and the source ontology uploader. One should refer to the Whyis documentation for further information about these features, which will allow a user to potentially automate usage of these features.

Based on the established and enhanced workflow, we can apply minor adjustments to support real-life use cases. The first potential use case is a browser and knowledge discovery system. In this case we utilize the newly created inference agents in Whyis to explore the knowledge created from our information extraction output. Assertions might provide information about an assertion discovered from related interactions sharing similar entities. A second potential use case an interaction question answer system. An application of such a system may be a domain expert trying to determine if two entities have an association with one another and the PubMed source where the association is defined. Overall, the infrastructure currently in development and the portions of the workflow already implemented provide several opportunities for future studies in the biomedical domain.

Chapter 7
FUTURE WORK AND CONCLUSION

The workflow implementation and evaluation has provided several opportunities for future work. The issues pertaining to the Liberal Event Extraction implementation are currently being alleviated. Advanced work pertaining to the Liberal Event Extraction includes evaluation of the AMR portion in relation to the event extraction portion and clustering of frame data to determine specific interactions from the Liberal Event Extraction output. The evaluation of Reach output based on fact matching accuracy can be expanded with inferencing within Whyis by aligning our results with its source ontologies, confidence derivation, and document provenance expansion. The workflow portions for Reach and Liberal Event Extraction can be implemented into Whyis for future projects using a similar methodology to produce long-term sustainability of the completed work within this thesis. Improvements in the knowledge conversion process for nanopublication creation can be made to further standardize the templating process. Incorporation of knowledge inconsistency detection provides validation of RDF assertion data. Specific use cases alongside this workflow can further prove usages for real-world knowledge processing tasks.

In summary, we have implemented a workflow using information extraction tools to extract knowledge from unstructured documents and convert the output to RDF graphs. This process offers an automated method of deriving domain-specific assertions, as opposed to employing manually created assertions for large-scale heterogeneous knowledge graph construction. Each information extraction tool has associated benefits and disadvantages when considering its workflow construction, initial knowledge representation, and external reference mapping. The extensible and flexible nature of the Whyis nano-scale knowledge graph framework provides a essential infrastructure for publishing, management and analysis of our information exaction workflow output. Our evaluation results provide substantial evidence towards knowledge alignment with the aforementioned manually derived annotations and potential steps towards probabilistic based knowledge inferencing from confidence-based metrics and source ontologies.

REFERENCES

[1] A. L. L. Phyu and N. Thein, "Domain adaptive information extraction using link grammar and wordnet," in *Fifth International Conference on Creating, Connecting and Collaborating through Computing (C5 07)*. IEEE, Jan. 2007, pp. 47–53.

[2] L. Huang, J. May, X. Pan, H. Ji, X. Ren, J. Han *et al.*, "Liberal entity extraction: Rapid construction of fine-grained entity typing systems," *Big Data*, vol. 5, no. 1, pp. 19–31, Mar. 2017.

[3] W3C, "Semantic web," https://www.w3.org/standards/semanticweb/, accessed on: Feb. 02, 2018.

[4] M. A. Valenzuela-Escarcega, G. Hahn-Powell, T. Hicks, and M. Surdeanu, "A domain-independent rule-based framework for event extraction," in *Proceedings of ACL-IJCNLP 2015 System Demonstrations*. Association for Computational Linguistics (ACL), Jul. 2015, pp. 127–132.

[5] L. Huang, T. Cassidy, X. Feng, H. Ji, C. R. Voss, J. Han *et al.*, "Liberal event extraction and event schema induction," in *ACL*, 2016.

[6] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., May 16, 2008.

[7] P. Groth, A. Gibson, and J. Velterop, "The anatomy of a nanopublication," *Information Services & Use*, vol. 30, no. 1-2, pp. 51–56, Jan. 2010.

[8] J. McCusker, "Whyis: a nano-scale knowledge graph framework," https://tetherless-world.github.io/whyis/, 2018, accessed on: Mar. 23, 2018.

[9] S. Atdag and V. Labatut, "A comparison of named entity recognition tools applied to biographical texts," *2nd International Conference on Systems and Computer Science*, vol. abs/1308.0661, pp. 228–233, Aug. 2013.

[10] J. G. Zheng, D. Howsmon, B. Zhang, J. Hahn, D. McGuinness, J. Hendler *et al.*, "Entity linking for biomedical literature," May 2015.

[11] Mausam, M. Schmitz, R. Bart, S. Soderland, and O. Etzioni, "Open language learning for information extraction," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, ser. EMNLP-CoNLL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, Jul. 2012, pp. 523–534.

[12] H. Cunningham, "Gate, a general architecture for text engineering," *Computers and the Humanities*, vol. 36, no. 2, pp. 223–254, May 2002, [Online]. Available: https://doi.org/10.1023/A:1014348124664

[13] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60.

[14] S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python*. OReilly Media Inc., Jun. 2009.

[15] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob *et al.*, "Abstract meaning representation for sembanking," in *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 178–186.

[16] M. A. Valenzuela-Escarcega, O. Babur, G. Hahn-Powell, D. Bell, T. Hicks, E. Noriega-Atala *et al.*, "Large-scale automated reading with reach discovers new cancer driving mechanisms," in *Proceedings of the Sixth BioCreative Challenge Evaluation Workshop*, 2017, pp. 201–203.

[17] M. A. Valenzuela-Escárcega, G. Hahn-Powell, and M. Surdeanu, "Description of the odin event extraction framework and rule language," *CoRR*, vol. abs/1509.07513, Sep. 2015.

[18] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, 2nd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., May 20, 2011.

[19] W3C, "Prov-dm: The prov data model," https://www.w3.org/TR/prov-dm/, 2018, accessed on: Mar. 07, 2018.

[20] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web*, vol. 8, pp. 489–508, 2016.

[21] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation, and Applications*. New York, NY, USA: Cambridge University Press, 2003.

[22] L. E. Hunter, "Knowledge-based biomedical data science," *Data Science*, vol. 1, pp. 19–25, Mar. 2017.

[23] S. M. Rashid, A. Viswanathan, I. Gross, E. F. Kendall, and D. L. McGuinness, "Leveraging semantics for large-scale knowledge graph evaluation," 2017.

[24] X. Wilcke, P. Bloem, and V. De Boer, "The knowledge graph as the default data model for machine learning," *Data Science*, pp. 1–19, Oct. 2017.

[25] L. Huang, H. Ji, K. Cho, and C. R. Voss, "Zero-shot transfer learning for event extraction," *CoRR*, vol. abs/1707.01066, Jul. 2017.

[26] F. Hogenboom, F. Frasincar, U. Kaymak, F. de Jong, and E. Caron, "A survey of event extraction methods from text for decision support systems," *Decision Support Systems*, vol. 85, pp. 12–22, May 2016.

[27] J. P. McCusker, M. Dumontier, R. Yan, S. He, J. S. Dordick, and D. L. McGuinness, "Finding melanoma drugs through a probabilistic knowledge graph," *PeerJ Computer Science*, vol. 3, p. e106, Feb. 2017.

[28] J. Pujara, H. Miao, L. Getoor, and W. Cohen, "Knowledge graph identification," in *The Semantic Web – ISWC 2013*, H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, Oct. 2013, pp. 542–557.

[29] Z. Huang, J. Yang, F. van Harmelen, and Q. Hu, "Constructing knowledge graphs of depression," in *Health Information Science*, S. Siuly, Z. Huang, U. Aickelin, R. Zhou, H. Wang, Y. Zhang *et al.*, Eds. Cham: Springer International Publishing, Oct. 11, 2017, pp. 149–161.

[30] M. Wang, J. Zhang, J. Liu, W. Hu, S. Wang, X. Li *et al.*, "Pdd graph: Bridging electronic medical records and biomedical knowledge graphs via entity linking," *CoRR*, vol. abs/1707.05340, Jul. 2017.

[31] X. Zhao, Z. Xing, M. A. Kabir, N. Sawada, J. Li, and S. W. Lin, "Hdskg: Harvesting domain specific knowledge graph from content of webpages," in *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, Feb. 2017, pp. 56–67.

[32] K. Gandhi and N. Madia, "Information extraction from unstructured data using rdf," in *2016 International Conference on ICT in Business Industry Government (ICTBIG)*. IEEE, Nov. 2016, pp. 1–6.

[33] National Center for Biotechnology Information, "Open access subset," https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/, Jul. 2018, accessed on Mar. 13, 2018.

[34] M. Valenzuela, G. Hahn-Powell, D. Bell, T. Hicks, E. Noriega, and M. Surdeanu, "Reach biomedical information extraction," https://github.com/clulab/reach, 2018, accessed on Mar. 13, 2018.

[35] J. Flanigan, S. Thomson, J. Carbonell, C. Dyer, and N. A. Smith, "A discriminative graph-based parser for the abstract meaning representation," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 1426–1436.

[36] C. Wang, N. Xue, and S. Pradhan, "A transition-based algorithm for amr parsing," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, May–Jun. 2015, pp. 366–375.

[37] M. Valenzuela, G. Hahn-Powell, D. Bell, T. Hicks, E. Noriega, and M. Surdeanu, "bioresources," https://github.com/clulab/bioresources/wiki/KBs, 2018, accessed on Mar. 13, 2018.

[38] S. Razick, G. Magklaras, and I. M. Donaldson, "irefindex: A consolidated protein interaction database with provenance," *BMC Bioinformatics*, vol. 9, no. 1, p. 405, Sep. 2008.

[39] S. Harris, A. Seaborne, and E. Prud'hommeaux, "Sparql 1.1 query language," https://www.w3.org/TR/2013/REC-sparql11-query-20130321/, W3C, Mar. 2013, accessed on Mar. 13, 2018.

[40] UniProt, "Uniref," https://www.uniprot.org/help/uniref, UniProt Consortium, Sep. 2017, accessed on Mar. 28, 2018.

Appendix A
MENTION TYPE GROUNDING

Below shows an example of interaction types, trigger types, and entity types produced by the Reach and Liberal Event Extraction methodologies. These types are grounded to linked data references in related biomedical ontologies. The prefixes in the table having the following URIs:

- Semanticscience Integrated Ontology (SIO): <http://semanticscience.org/resource/SIO_>
- Gene Ontology (GO): <http:purl//obolibrary.org/obo/GO_>
- Molecular Interactions (MI): <http://purl.obolibrary.org/obo/MI_>
- NCI Thesaurus (NCIT): <http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#>
- Cell Ontology (CL): <http://purl.obolibrary.org/obo/CL_>

Table A.1. Concept to class sample groundings for biomedical interaction/association typed events and their associated arguments for both information extraction methodologies. Includes groundings for entity types.

	Concept	**Class**
1	acetylation	mi:0192; go:0006473
2	activation	mi:2254; go:0072376
3	amino-acid	sio:0192
4	amount	sio:000052; ncit:C25256
5	autophosphorylation	go:0046777
6	bind	go:0005488
7	bioprocess	mi:0359;go:0008150
8	cellline	sio:010054
9	cell-type	sio:010001; cl:0000000
10	cellular-component	mi:0354; go:0005575
11	complex-assembly	go:0006461

12	controlled	ncit:C61299
13	controller	sio:000879
14	deacetylation	mi:0197; go:0006476
15	deglycosylation	mi:0558; go:0006517
16	demethylation	mi:0871; go:0070988
17	dephosphorylation	mi:0203; go:0016311
18	destination	sio:010423
19	enhance	ncit:C71607
20	enzyme	mi:0501
21	family	sio:001380; ncit:C45290
22	farnesylation	mi:0206; go:0018343
23	gene-or-gene-product	mi:0251
24	glycosylation	mi:0559; go:0070085
25	hydrolysis	go:0016787
26	hydroxylation	mi:0210; go:0018126
27	induce	ncit:C61367
28	inhibit	mi:0586
29	methylation	mi:0213; go:0032259
30	negative-activation	go:0050866
31	negative-regulation	go:2000258
32	organ	sio:010003; uberon:0000062
33	phosphorylation	mi:0217; go:0016310
34	positive-activation	go:0050867
35	positive-regulation	go:2000259
36	potentiate	mi:2237
37	protein	mi:0326; sio:010043
38	protein-modification	go:0036211
39	regulation	go:2000257
40	ribosylation	mi:0557; go:0006471
41	simple-chemical	sio:010004

42	site	sio:000019
43	source	ncit:C25683; sio:000253
44	species	ncit:C45293
45	sumoylation	mi:0566; go:0016925
46	theme	sio:000798
47	tissuetype	sio:010002
48	transcription	sio:010300; ncit:C17208
49	translocation	mi:0593; go:0051836
50	ubiquitination	mi:0220; go:0016567

Appendix B
SAMPLE INFORMATION EXTRACTION OUTPUT

We provide sample output from the both information extraction methodologies to better illustrate the difference between their output approach. Then, we standardize the Liberal Event Extraction output to the FRIES format in Figure 4.4 and convert the output to a nanopublication format.

```
1  When compared to p38 SAPK, MAPK1 was clearly able to phosphorylate the ASPP2
       fragment in vitro (Figure 1B, left and middle panels). 3
2  ##Event: bio-bmtr_0005##11##0.2.0####phosphorylate-01 phosphorylate
3  ##Argument: bio-bmtr_0005##11##0.1.0####protein-segment :ARG1 Patient
4  ##Argument: bio-bmtr_0005##11##0.2.0.0####enzyme :ARG2 Patient
5
6  ##Event: bio-bmtr_0005##11##0.1####phosphorylate-01 phosphorylate
7  ##Argument: bio-bmtr_0005##11##0.0##clearly##clear :mod Explanation
8  ##Argument: bio-bmtr_0005##11##0.1.1####enzyme :ARG2 Patient
9  ##Argument: bio-bmtr_0005##11##0.1.0####protein-segment :ARG1 Patient
10 ##Argument: bio-bmtr_0005##11##0.1.2####in-vitro :manner Manner
11
12 ##Event: bio-bmtr_0005##11##0.3####describe-01 lead
13 ##Argument: bio-bmtr_0005##11##0.3.0##and##and bio-bmtr_0005
       ##11##0.3.0.2.0####"1B" :ARG0 agent
14 ##Argument: bio-bmtr_0005##11##0.0##clearly##clear :mod Explanation
15 ##Argument: bio-bmtr_0005##11##0##able##possible :ARG1 theme
```

(a) Sample Liberal Event Extraction output.

```
{
    "frame-id" :"evem-PMC1459871-UAZ-r1-1459871-87-34",
    "text" :"interaction between PIG-2 and YWHAH",
    "arguments" :[{
      "text" :"PIG-2",
      "argument-type" :"entity",
      "type" :"theme",
      "object-type" :"argument",
      "index" :0,
      "arg" :"ment-PMC1459871-UAZ-r1-1459871-87-316"
    },{
      "text" :"YWHAH",
      "argument-type" :"entity",
      "type" :"theme",
      "object-type" :"argument",
      "index" :1,
      "arg" :"ment-PMC1459871-UAZ-r1-1459871-87-317"
    }],
    "type" :"complex-assembly",
    "frame-type" :"event-mention",
    "is-direct" :true,
    "end-pos" :{
      "reference" :"pass-PMC1459871-UAZ-r1-1459871",
      "offset" :13496,
      "object-type" :"relative-pos"
    },
    "trigger" :"interaction",
    "object-type" :"frame",
    "start-pos" :{
      "reference" :"pass-PMC1459871-UAZ-r1-1459871",
      "offset" :13461,
      "object-type" :"relative-pos"
    },
    "sentence" :"sent-PMC1459871-UAZ-r1-1459871-87",
    "found-by" :"binding11",
    "context" :["cntx-PMC1459871-UAZ-r1-1459871-86-34" ],
    "verbose-text" :"To investigate the domain of interaction between
       PIG-2 and YWHAH, we have used four GST fusion constructs, YWHAH (1
       -60), YWHAH (1-80), ..."
}
```


(b) Sample Reach output.

Figure B.1. Sample output from both Information Extraction methodologies.