COMPREHENSIVE DEEP LEARNING PIPELINE FOR WHALE SHARK RECOGNITION

Maksim Kholiavchenko

Submitted in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

Approved by:
Dr. Charles Stewart, Chair
Dr. Alex Gittens
Dr. Radoslav Ivanov

Department of Computer Science
Rensselaer Polytechnic Institute
Troy, New York

[May 2022]
Submitted March 2022

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The whale shark is the largest fish species in existence today. The main threat to the whale shark population is poaching. Despite conservation efforts, whale shark hunting persists in tropical countries due to population increase and, as a result, growing demand for food. The long maturation period and slow rate of reproduction add to the whale shark population's vulnerability. Whale sharks are listed as endangered species by the International Union for Conservation of Nature, which estimates a 50% decline in the whale shark population over the last 75 years.

Whale sharks migrate over great distances in search of plankton. To date, little is known about whale sharks' life cycle, characteristics of their behavior, and reproduction. Recognition of whale sharks is a starting point for studying the migrations of these animals. In this work, we present an approach for whale shark recognition through a region of interest detection, spot segmentation, and deep metric learning. Whale sharks are speckled with dazzling white spots and lines. Such natural markings are distinctive which makes it possible to achieve good recognition results with modern deep learning techniques.

In this work, we employ a multi-stage approach to tackle the problem of whale shark recognition. Firstly, we prepare a novel whale shark detection dataset and train the YOLOv5s model to detect areas from the pectoral fin to the dorsal fin. This area contains a large amount of whale shark biometric information such as uniquely patterned white spots. Secondly, we train a U-net model with the SEResNet34 backbone to segment these spots on whale sharks' bodies. Thirdly, we train an InceptionResNet embedding model which makes use of spots location as well as originally detected whale shark image to produce high-quality embedding. Finally, we introduce an embedding-based recognition algorithm and validate its performance. For the experiment without new individuals in the test set, our algorithm scores 93% Top-1 recognition accuracy, while for the experiment with new individuals in the test set, it scores 83%.

1. INTRODUCTION

1.1 Problem Overview

Whale shark populations around the world are constantly declining. Modern computer vision techniques are an excellent tool for tracking whale shark migrations. A whale shark encounter and subsequent identification can also assist us in determining whether the animal is alive. Whale sharks have a distinct appearance that acts as biometric information, allowing us to precisely identify an animal. An example of a whale shark's unique dotted pattern is shown in Figure 1.1.

Figure 1.1: Whale sharks are speckled with dazzling white spots and lines

In this work, we investigate modern approaches to detection, segmentation, and metric learning for the recognition of whale sharks. Over the last decade, computer vision has advanced significantly, and modern deep neural networks outperform humans in many tasks [1] – [3]. Object detection is the problem of identifying and locating objects in an image. Object detection helps in our task by filtering out irrelevant views and detecting a specific region of interest for further processing. As soon as we have an appropriate region of interest, we can use segmentation to locate the white spots on the animal's body. Then, we apply a deep convolutional network to extract features from the image and spot locations. Finally, we utilize a

ranking algorithm to perform recognition using computed features. The ranking algorithm uses embeddings to find the best matches for the requested whale shark image. It can generate the Top-N closest whale sharks in a database or give a decision that the requested whale shark is not in a database.

1.2 Whale Shark Dataset for Detection and Deep Metric Learning

To detect the area from the pectoral fin to the dorsal fin and solve the whale shark recognition problem, we need a specific dataset to train deep neural networks. For this purpose, we extend the whale shark identification dataset [4]. The original dataset contains images and labels in Microsoft COCO format. An example of an image from the dataset is shown in Figure 1.2.

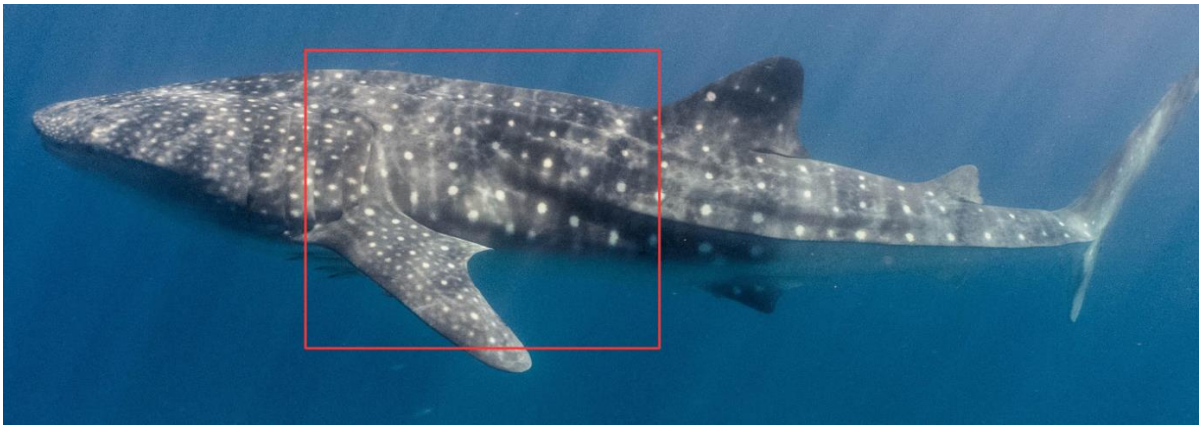

Figure 1.2: An image from the original dataset and manually labeled bounding box

Each image is accompanied by a bounding box of the whale shark's entire body, an individual identification tag, and a viewpoint of the animal. A total of 7693 named sightings are reported for the 543 individual whale sharks. The main issue with the original dataset is that it contains a lot of inappropriate views, such as only the tail visible, only the back of a shark visible, or animals are half-rotated. Figure 1.3 shows examples of inappropriate whale shark views.

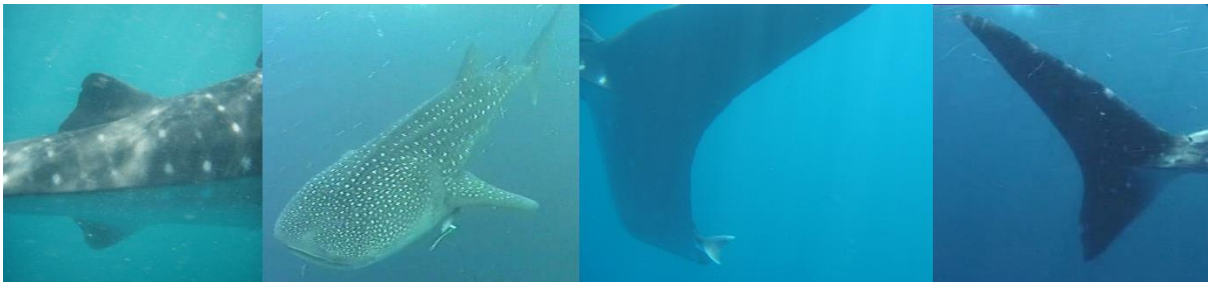

Figure 1.3: Examples of inappropriate whale shark views

Furthermore, it contains images of small fishes covering the whale shark's body, making the location of the spots indistinguishable. To tailor the dataset for our task, we filter out all images labeled "back" for the viewpoint, filter out all individuals that contain less than 2 images per animal, and manually annotate all remaining images with bounding boxes of the area from the pectoral fin to the dorsal fin. It is important to have enough images for each individual to be able to train an identification model.

After processing, our dataset consists of 3424 named sightings for the 625 individual whale sharks. To train an identification model, we treat left-positioned images and right-positioned images of the same whale shark as different individuals. Horizontal flip is applied to right-positioned images. The distribution of images with good viewpoints and bad viewpoints is shown in Figure 1.4. Bad viewpoints were intentionally left in the dataset to potentially reduce false positive detections.

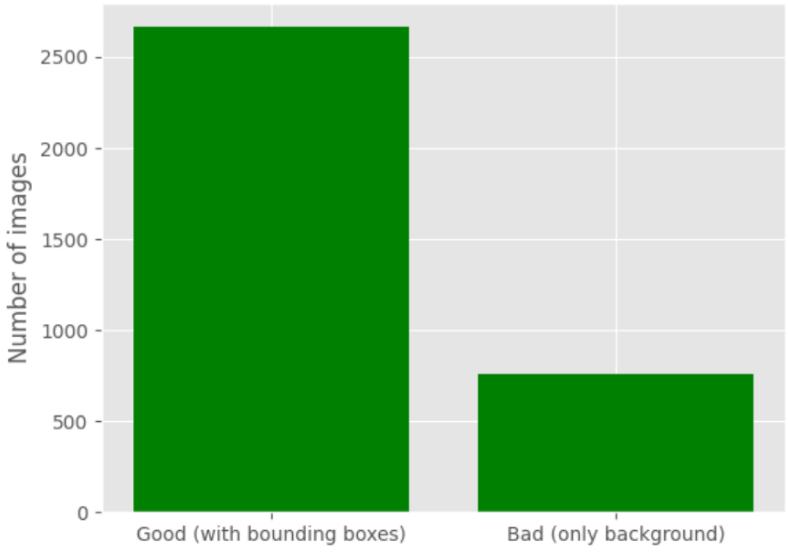

Figure 1.4: Number of images for good and bad viewpoints

Only images taken from left or right viewpoints are included in the processed dataset. In addition, to maintain consistency for an embedding network, right-positioned images were mirrored. Left-view and right-view images of the same whale shark are treated as distinct classes. The distribution of images per individual whale shark is depicted in Figure 1.5.

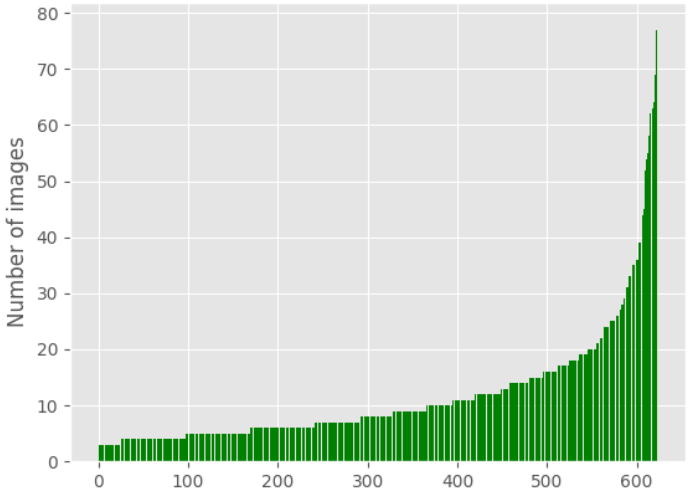

Figure 1.5: Number of images per individual whale shark

1.3 Whale Shark Spot Segmentation Dataset

The whale shark spot segmentation dataset was provided by Wildbook for Sharks [5]. The original dataset consists of whale shark images and metadata such as spot locations and individual identification tags. The dataset provides annotations for spot locations only for the area from the pectoral fin to the dorsal fin. Because the dataset was labeled by multiple annotators, the quality of annotations may be inconsistent. An example of a whale shark image and spot locations is shown in Figure 1.6.

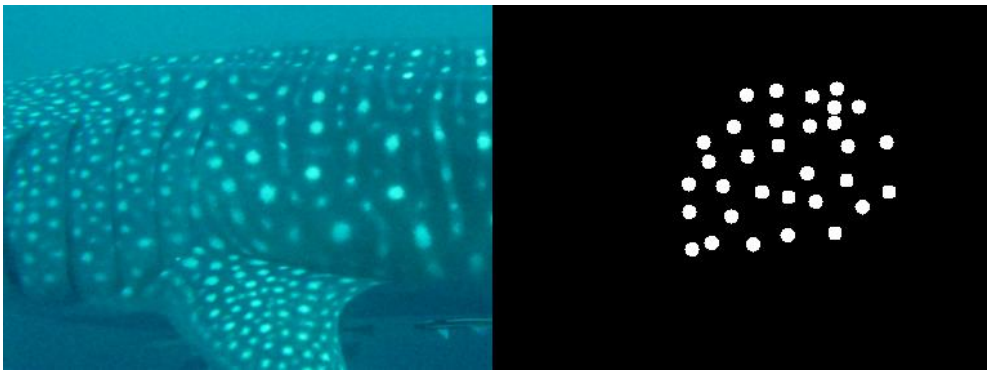

Figure 1.6: An example of a whale shark image and spot locations

A total of 2474 images are provided for the 1537 individual whale sharks. Most distinct whale shark identities appear only once in the dataset and for 269 images there was no individual identification tag provided. The distribution of whale shark sightings is depicted in Figure 1.7.

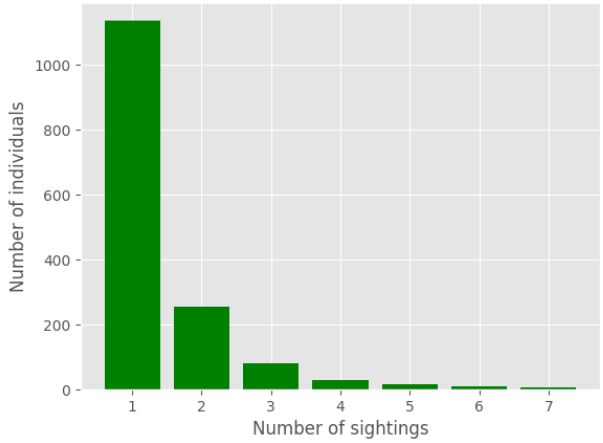

Figure 1.7: Number of images per individual whale shark

2. LITERATURE REVIEW

2.1 Algorithms for Animal Identification

The development of computer vision technologies has enabled the solution of an animal identification problem. The use of computer vision for facial recognition has already become a global industry standard. These technologies are used by large corporations such as Apple, Facebook, and Google to identify users and grant access to data and bank accounts. With the help of computer vision, it became possible to use the appearance of a person for re-identification [6], which can be useful for applications such as multi-person tracking [7], [8]. For reliable identification, we can only rely on biometric identifiers. Biometric identifiers are distinguishing, quantifiable characteristics that are used to label and describe individuals [9]. Animals, unlike humans, do not wear clothes, and many species have distinct color patterns, allowing their bodies to be used for full identification.

The authors of [10] came up with an algorithm for the sea turtle's identification. It turned out that hawksbill turtles have a distinctive pattern all over their heads. The authors used this knowledge to create an algorithm that utilizes SIFT [11] features for identification.

Wildbook, an autonomous computational system that uses computer vision to locate and identify individual animals of mostly striped, spotted, wrinkled, or notched species, was introduced by the authors of [12]. Flukebook, an open-source cetacean data archiving and photo-identification tool was developed as part of this platform [13].

The authors of [14] investigate a humpback whales identification approach by applying a propriety photo-identification matching system based on a discrete categorization of ventral-tail flukes. The system was based on an array of coded discrete characteristics.

The authors of [15] propose a method for animal identification against a labeled dataset. The authors used hot spots of an animal appearance to find correspondences between two images The algorithm was tested for Grevy's zebras, plains zebras, giraffes, leopards, and lionfish.

The most related work for our domain is an astronomical pattern-matching algorithm that was applied for whale sharks' identification [16]. The authors annotated whale sharks' spots and using this information, applied Groth's algorithm [17] to match triangle pairs produced for these spots. The method proved to be extremely accurate, and it is now included in the open-source library Wildbook for Whale Sharks [5].

2.2 Convolutional Neural Networks for Object Detection

Object detection has been one of the most useful applications of computer vision. Object detection is a computer vision approach for detecting locations and labels of objects in an image. Object detection approaches often use deep learning and convolutional neural networks. These approaches can be divided into two categories: two-stage-based solutions and one-stage-based solutions.

Two-stage networks identify region proposals or subsets of the picture that may contain an object. Then, two-stage networks classify the objects within the region proposals. The neural network predicts the coordinates of the object's bounding box and the class of the object inside the bounding box. Examples of the two-stage detectors are R-CNN [18], Fast R-CNN [19], Faster R-CNN [20], Cascade R-CNN [21].

One-stage networks produce predictions for regions throughout the entire image using anchor boxes, which are then decoded to form the final bounding boxes for the objects. Examples of the two-stage detectors are YOLO [22] – [27], EfficientDet [28], RetinaNet [29], CenterNet [30].

A good example of using a one-stage network for animal detection is the work [31] that presents a five-component detection. The authors employ image classification to determine the presence of animals in an image. Then, the authors use a YOLO-based network to perform annotation localization and place bounding boxes over the animals. Annotation localization is followed by annotation classification which provides information about species and viewpoint labels to each annotation. After that, annotation background segmentation creates a foreground-background mask for each species. Finally, the annotation of interest classification predicts the image's focus.

In this work, we adapt YOLOv5s architecture for the detection step of our pipeline. This model proves to be both fast and accurate while requiring reasonable training time.

2.3 Convolutional Neural Networks for Semantic Segmentation

Semantic segmentation is the problem of assigning a class label to each pixel in an image. Semantic segmentation, as opposed to detection, allows for more accurate localization of objects in an image. Over the last decade, deep convolutional neural networks have performed admirably in semantic segmentation tasks. Semantic segmentation is now widely employed in self-driving cars [32], [33], computational photography [34], satellite imaging [35], and medical imaging [36], [37]. Images and accompanying masks must be prepared to train the model for semantic segmentation. Masks are typically represented as binary matrices with non-zero values for the targeted pixels and zeros for the background.

The most used metrics for semantic segmentation are the intersection-over-union (IoU), also known as the Jaccard index (Equation 2.1), and the Dice coefficient (Equation 2.2) [38].

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \tag{2.1}$$

Both metrics are statistical approaches to assess the similarity between two sets of data but only the Dice coefficient can be used in a loss function because it is differentiable.

$$D(X,Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{2.2}$$

U-Net [39] architecture is one of the most often used neural network topologies for semantic segmentation. This architecture was first applied to biomedical images. It consists of two parts: an encoder network and a decoder network. The encoder network performs downsampling and encodes input images to feature representation. The decoder network often mirrors encoder architecture and performs upsampling from feature representation to output mask. The central aspect of U-Net architecture is that the decoder also uses concatenation operation to forward information from the respective layers of the encoder network. It is worth mentioning that VGG [40], ResNet [41], SE-ResNet [42], ResNeXt [43], and other architectures can be used for the encoder network.

LinkNet [44] is another example of a neural network architecture for semantic segmentation. The authors modified U-Net architecture and made it faster by replacing concatenation operation with addition operation.

DeepLab [45], unlike U-Net and LinkNet, uses a different approach and utilizes dilated convolutions [46] and pointwise convolutions [47] as the main feature to preserve image shape. This architecture also uses upsampling but only at the very end of the network.

In our experiments, we employ U-Net architecture with the SE-ResNet34 backbone.

2.4 Convolutional Neural Networks for Feature Extraction

Metric learning is a widely used strategy for human and animal identification. To compare distances between two images, such images are usually mapped into embeddings, low-dimensional vector representations. Metric learning is a method of determining the degree of similarity between images that are based solely on a distance metric.

There are several ways to produce embeddings. Classic methods include Principal Component Analysis (PCA) [48] and t-Distributed Stochastic Neighbor Embedding (t-SNE) [49]. Deep convolutional neural networks are now being utilized more frequently to encode images into embeddings. Deep convolutional neural networks are designed to learn hierarchical feature representations by building high-level features from low-level ones. Such approaches have proven to be effective in face recognition [50], [51], person re-identification [52] – [54], vehicle re-identification [55], and animal identification [56], [57].

It is usually necessary to use special loss functions to train such models. The authors of [50] came up with a triplet loss function that relies on a set of three images (an anchor image, a positive image, and a negative image). The purpose of triplet loss is to get the anchor embedding closer to all the positive sets of embeddings while keeping it far away from the negative sets. The authors of [58] presented a contrastive loss. The contrastive loss takes a positive embedding and calculates its distance to another embedding of the same class and contrasts that with the distance to negative embeddings. The authors of [59] introduced a center loss which adds a new regularization term to the SoftMax function [60]. This improvement allows

to simultaneously learn a center for each class and penalize the distances between the image embeddings and their corresponding class centers. The methods to measure the proximity of vectors in a vector space include cosine similarity and Euclidean distance.

In this work, we utilized a deep convolutional neural network that takes images and spot locations and generates embeddings for comparison.

3. WHALE SHARK DETECTION

The detection module was designed to find bounding boxes of the area from the pectoral fin to the dorsal fin of whale sharks. The detection of these locations is an important element of the pipeline. It serves two purposes: detection of the above-mentioned areas and filtering out bad viewpoints. Examples of inappropriate whale shark views are shown in Figure 1.3. It is essential to filter out images with bad viewpoints in the initial stage so that the segmentation in the next stage provides the expected results.

3.1 Data Augmentations

To increase the variability of the data, we used augmentations during the model training. Validation and testing did not involve augmented data. Colorspace augmentations such as HSV-Hue, HSV-Saturation, and HSV-Value were used. Vertical flip, perspective, shear, scale, rotation, and translation were among the affine augmentations used.

3.2 Model Architecture

To achieve high detection accuracy, we employed YOLOv5s [22] neural network architecture pre-trained on the COCO dataset. This network consists of a backbone, path aggregation network [61], and head. The backbone is primarily used to extract key features from input images. The path aggregation network is used to create feature pyramids which help the neural network to generalize better for object scaling. The head uses anchor boxes to produce final predictions of class probabilities, objectness scores, and bounding boxes. The loss function was taken directly from [22] with no modifications.

3.3 Model Training

The model was trained on 70% of the data, 10% was used for validation, and 20% was used for testing. With a batch size of 16 and an input image size of 512 by 512 pixels, the model was trained for 20 epochs. Stochastic gradient descent (SGD) optimizer with an initial learning rate of 0.01 was used. Then, a

learning rate scheduler was used to decrease the learning rate over time. The model was trained on NVIDIA GeForce RTX 3080. Figure 3.1 depicts the validation box loss plot and the validation object loss plot.

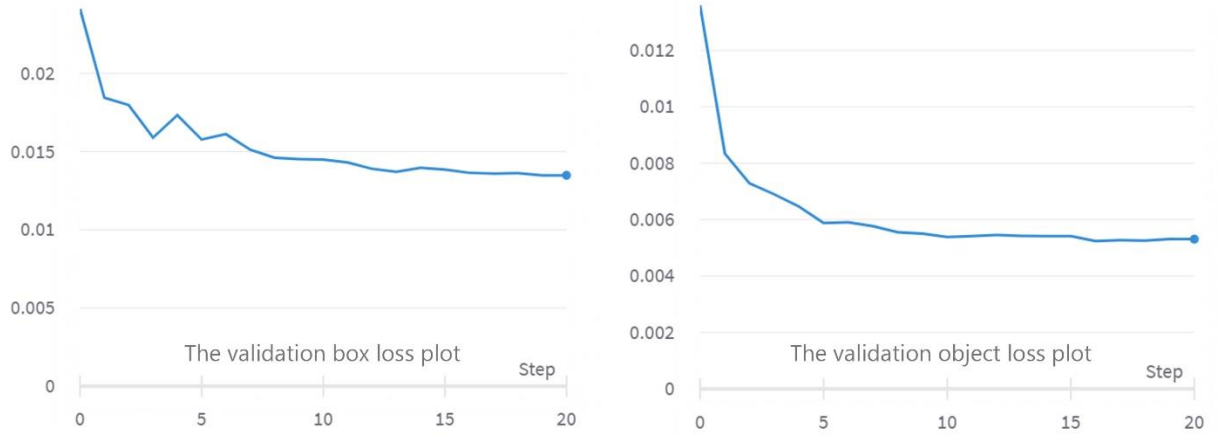

Figure 3.1: The validation box loss plot and the validation object loss plot

3.4 Results

Table 3.1 shows the results achieved by the proposed YOLOv5s network. The model achieved mean average precision of 0.995 at 0.5 IoU threshold. The horizontal flip was used as a test-time augmentation. These results demonstrate quite well performance for the detection of the area from the pectoral fin to the dorsal fin of whale sharks. It is worth mentioning that the quality of images and dataset annotations are quite good, and such conditions are not always easy to replicate in real life.

Table 3.1: Whale shark detection performance

Precision	Recall	mAP@0.5	mAP@0.5:0.95
0.993	0.994	0.995	0.704

Figure 3.2 shows some examples of the bounding box detections of the area from the pectoral fin to the dorsal fin of whale sharks. The middle right image does not display any bounding box because the view is not appropriate.

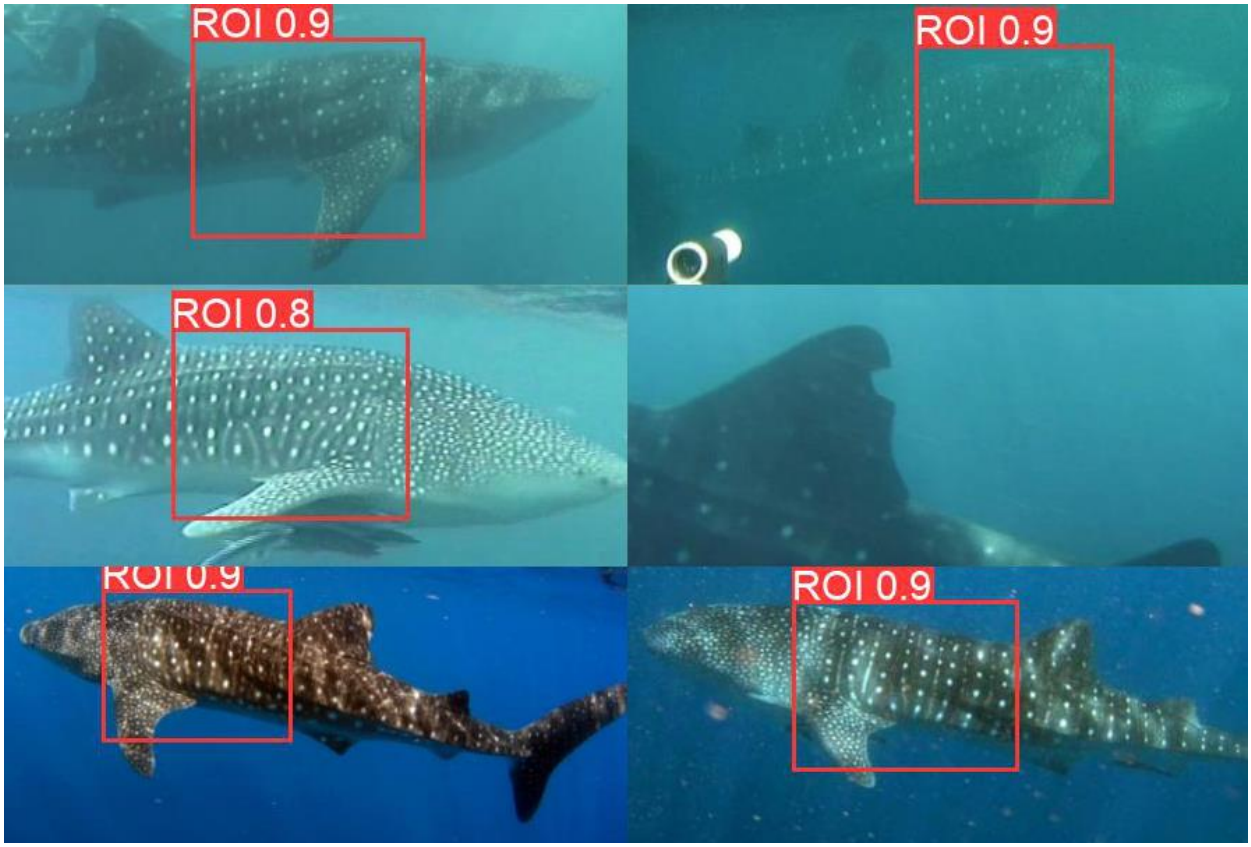

Figure 3.2: Examples of the model performance on test data

4. WHALE SHARK WHITE SPOT SEGMENTATION

Segmentation is used to locate white spots on the bodies of whale sharks. Image segmentation can be thought of as image classification at the pixel level. The neural network has to find all white spots in the whale shark image, as well as their precise location and boundaries at the pixel level. Examples of white spots segmentation are shown in Figure 4.1. The neural network generates binary masks in which each non-zero pixel represents a white spot's location. In Figure 4.1, all non-zero binary mask values are indicated in red.

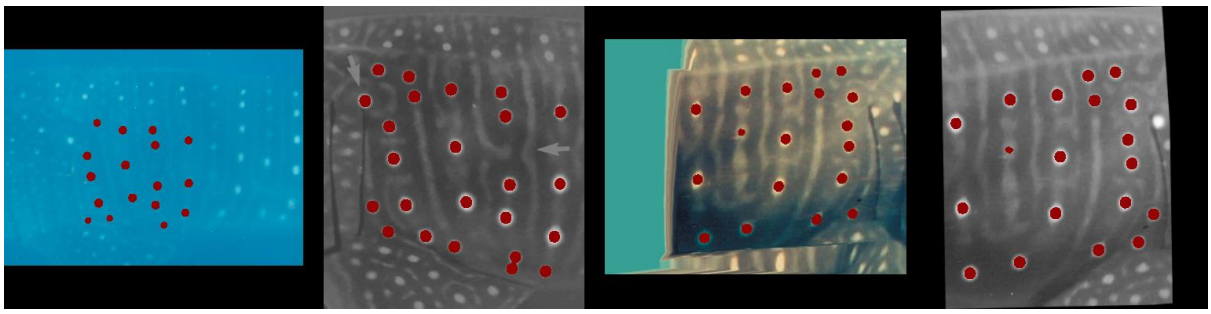

Figure 4.1: Examples of white spot segmentation

4.1 Data Augmentations

To artificially increase the amount of data to train a neural network for white spot segmentation, we employed augmentation techniques. The set of augmentations used is shown in Table 4.1.

Table 4.1: Data augmentations used to train the segmentation model

Augmentation	Parameters
Scale	from 80% to 120%
Rotate	from -15° to 15°
Shear	from -10° to 10°
Translate	from -10% to 10%
Horizontal Flip	50% of the time
Random Brightness and Contrast	20% of the time

The augmentations were applied both to the images and to the corresponding masks. Figure 4.2 displays examples of augmented data.

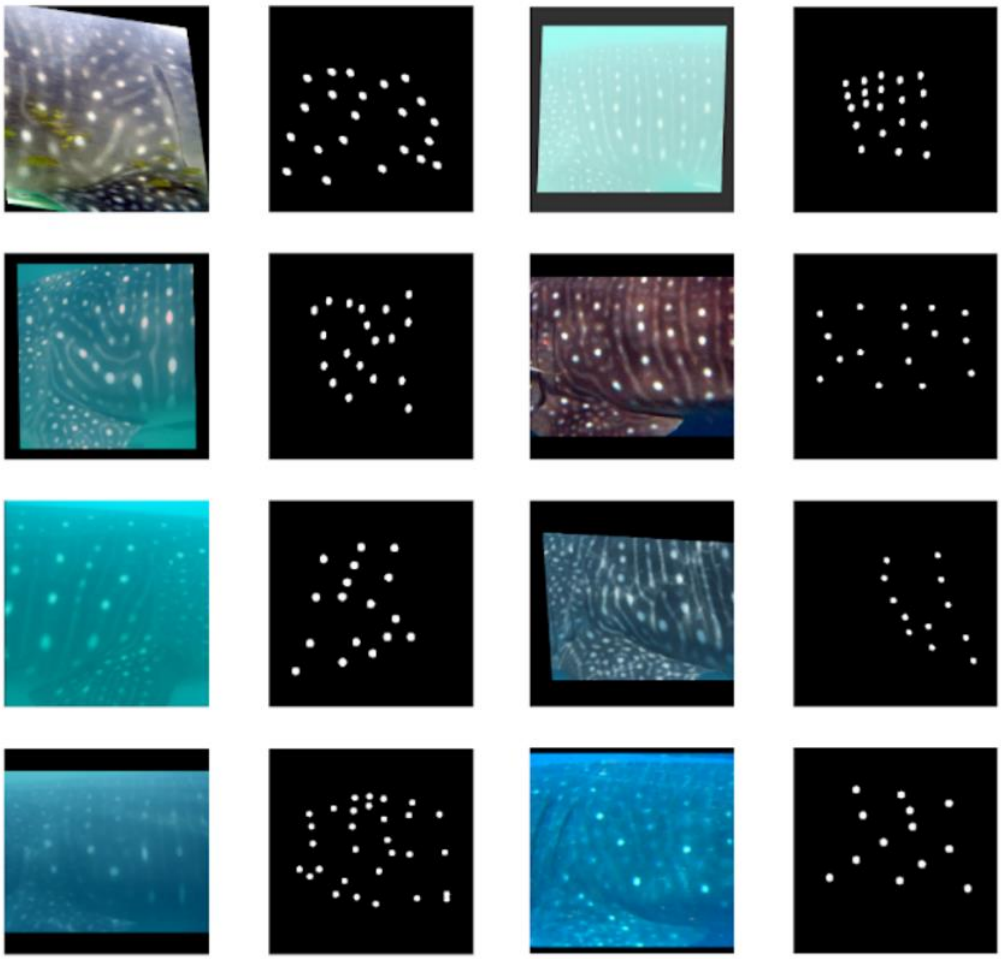

Figure 4.2: Examples of augmented images and masks

4.2 Model Architecture

The segmentation model follows the U-Net architecture with the SEResNet34 backbone. The U-Net architecture is symmetric and consists of an encoder part and a decoder part. An encoder part is a SEResNet34 network. This network is based on ResNet34 [41] but it uses squeeze-and-excitation blocks at the end of each non-identity branch of the residual blocks. A decoder part employs up-sampling layers to expand latent representation into an image of the original size. Encoder layers pass over information to the

corresponding decoder layers with the help of special connections. This helps to transfer the classification context to the localization part. The model architecture is shown in Figure 4.3.

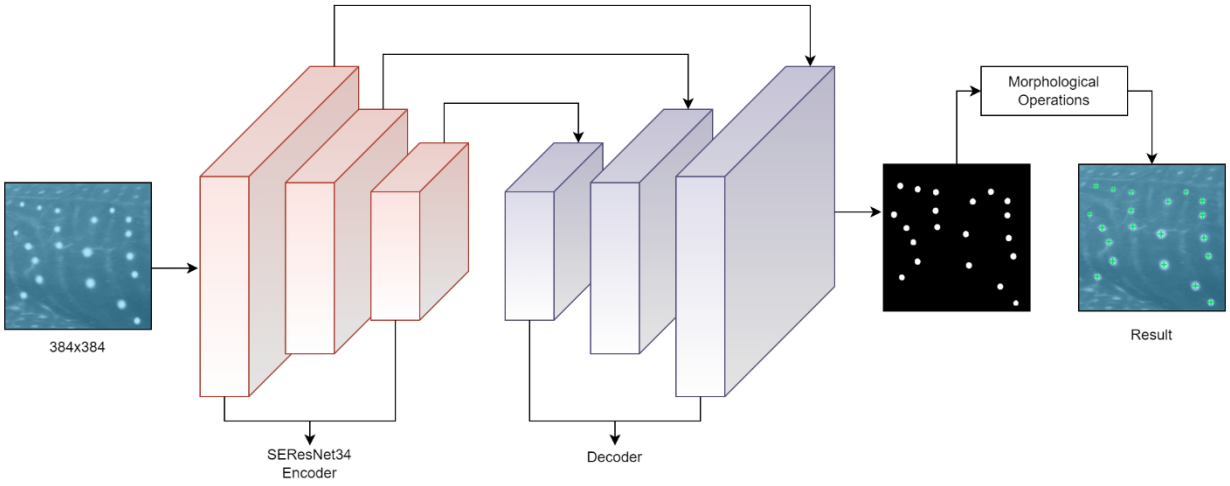

Figure 4.3: The U-net architecture with SEResNet34 backbone

The model produces a mask image. Then, the image is binarized by applying a threshold of 0.5. Morphological operations such as erosion and dilation are used to filter out very small blobs on the predicted mask. Finally, the centers of mass of the blobs are used to pinpoint the precise location of the segmented white spots of whale sharks.

4.3 Loss Function

To train our network, we utilize a two-component loss function. The loss function consists of the Dice loss function (Equation 4.1) based on the Dice coefficient [38] and binary cross-entropy loss function (Equation 4.2) that is commonly used for binary classification problems. The Dice loss function is defined as follows:

$$Dice(X, Y) = \frac{2 \sum_{p \in P} X_p Y_p}{\sum_{p \in P} X_p^2 + \sum_{p \in P} Y_p^2} \tag{4.1}$$

where X represents a predicted binary mask, Y represents a target ground-truth mask, and P represents a collection of pixel indexes in the provided mask. The binary cross-entropy loss function is defined as follows:

$$BCE(X, Y) = -\sum_{p \in P} Y_p \log X_p \tag{4.2}$$

where X represents a predicted binary mask, Y represents a target ground-truth mask, and P represents a collection of pixel indexes in the provided mask. The two-component loss function is defined as follows:

$$Loss(X, Y) = BCE(X_i, Y_i) - \log Dice(X_i, Y_i) \tag{4.3}$$

where X represents a predicted binary mask and Y represents a target ground-truth mask. The two-component loss function for the segmentation problem was explored by [62]. Such a combination takes the best of the two functions. The binary cross-entropy loss function helps to stably perform pixel-wise image classification, while the Dice loss function helps to handle effectively scenarios when the foreground area is small in comparison to the background area.

4.4 Model Training

The 4-folds cross-validation was used for the training protocol. The model was trained for 50 epochs with a batch size of 8 and an input image size of 384 by 384 pixels. Adam optimizer with an initial learning rate of 0.001 was used. Then, the learning rate was decreased each time the model reached a plateau. The model was trained on NVIDIA GeForce RTX 3080. Figure 4.4 depicts the train and validation Dice losses.

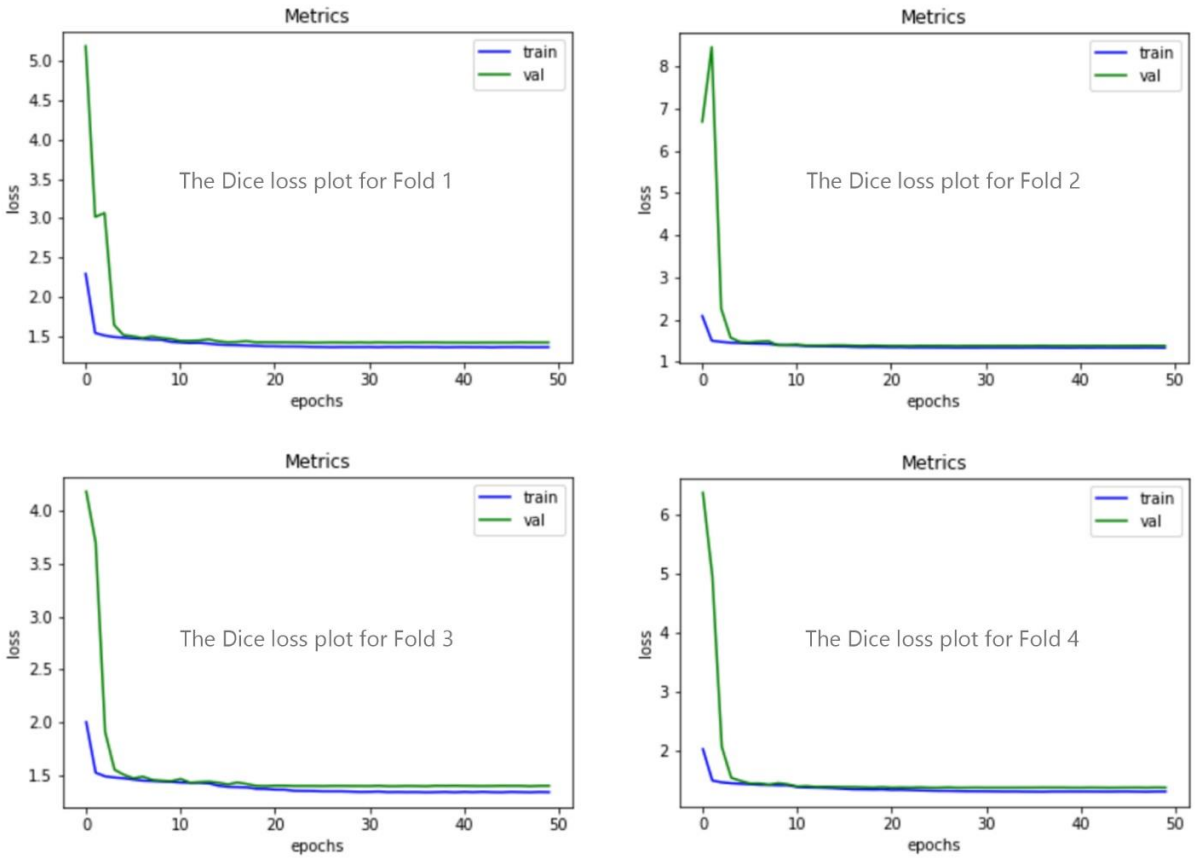

Figure 4.4: The train and validation Dice losses for U-net models

4.5 Results

The cross-validation results obtained by the U-Net model with the SEResNet34 backbone are shown in Table 4.2. The results were calculated for 4-folds and then averaged over the folds. The model achieved an average Jaccard coefficient of 0.770.

Table 4.2: Segmentation cross-validation results

Fold	Dice	IoU
1	0.739	0.795
2	0.672	0.754
3	0.711	0.774
4	0.681	0.756
Average	0.701	0.770

Figure 4.5 shows examples of cases that positively impact evaluation metrics. Green labels mean true-positive results, blue labels mean false-negative results and red labels mean false-positive results. The model does a good job of segmenting the white spots in the area above the pectoral fin. All false-positive segmentations and false-negative segmentations are white spots in the correct area but these white spots were not annotated in ground truth masks. This is due to inconsistencies in the ground truth data annotations.

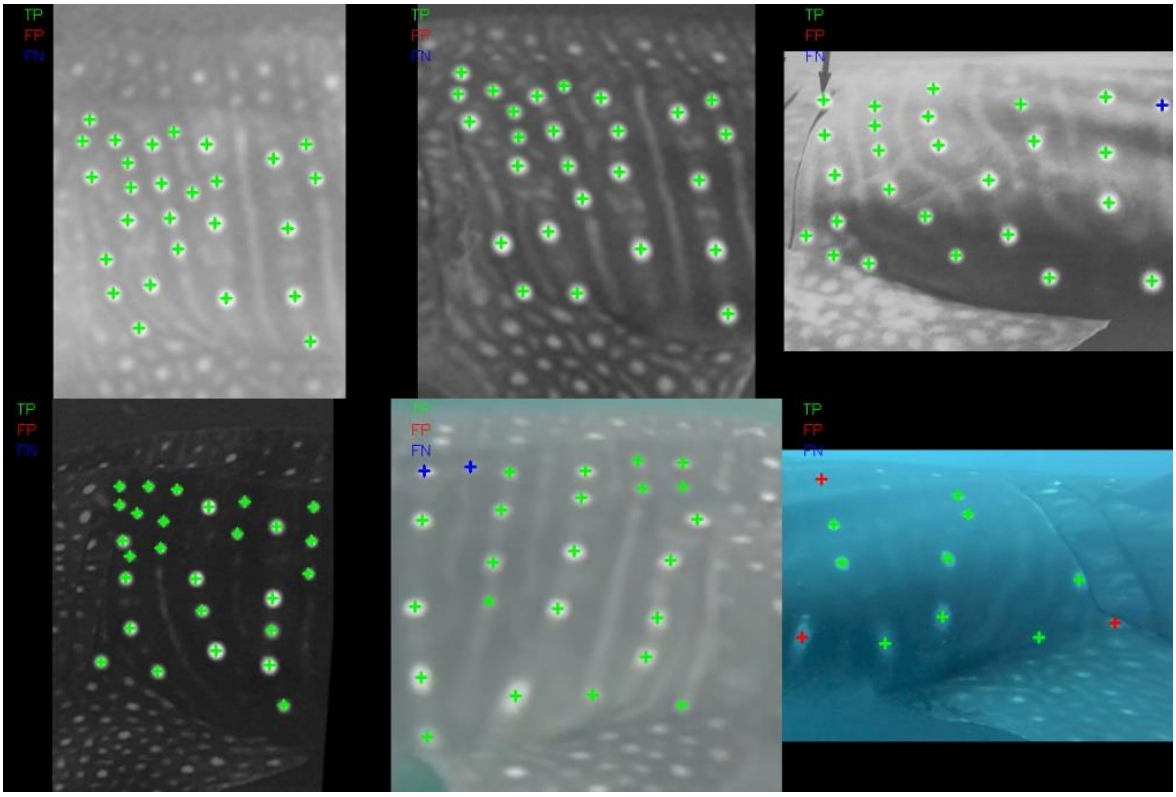

Figure 4.5: Examples of cases that positively impact evaluation metrics

Figure 4.6 depicts examples of cases that negatively impact evaluation metrics. Even though false-positive segmentations harm evaluation metrics, they are still useful for our task and they are labeled as false-positives mainly because the ground truth annotations are contradictory. Evaluation metrics are also negatively affected by errors in ground truth annotations. For example, image A comes with an empty ground truth mask, resulting in all segmented white spots being labeled as false positives.

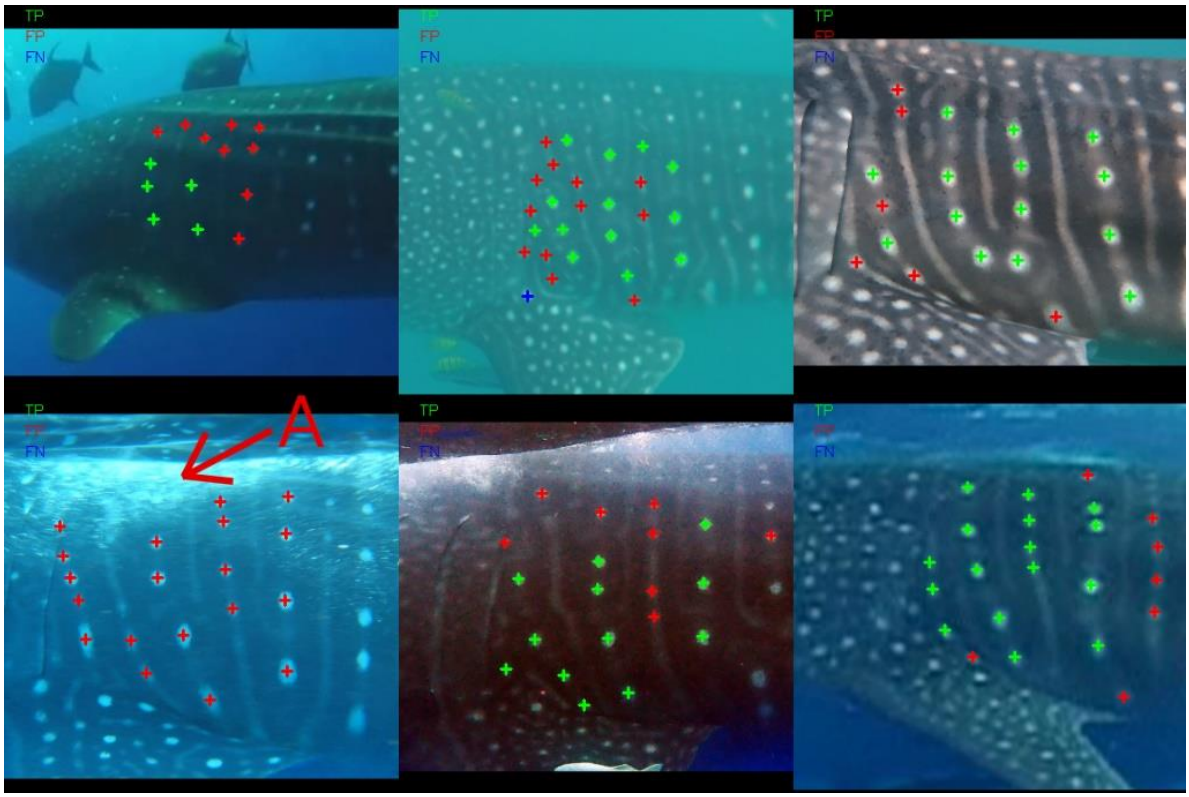

Figure 4.6: Examples of cases that negatively impact evaluation metrics

5. WHALE SHARK RECOGNITION

The whale shark recognition pipeline consists of several stages. Our algorithm takes a whale shark image as an input. The detection step is applied to locate the area from the pectoral fin to the dorsal fin of a whale shark. At this stage, our algorithm filters out inappropriate views. If nothing is detected, then the algorithm terminates and cannot proceed to the next stage. Then, the segmentation step is applied to the detected region. As a result, the segmentation step produces a binary mask of white spots on the whale shark's body. The whole pipeline is shown in Figure 5.1.

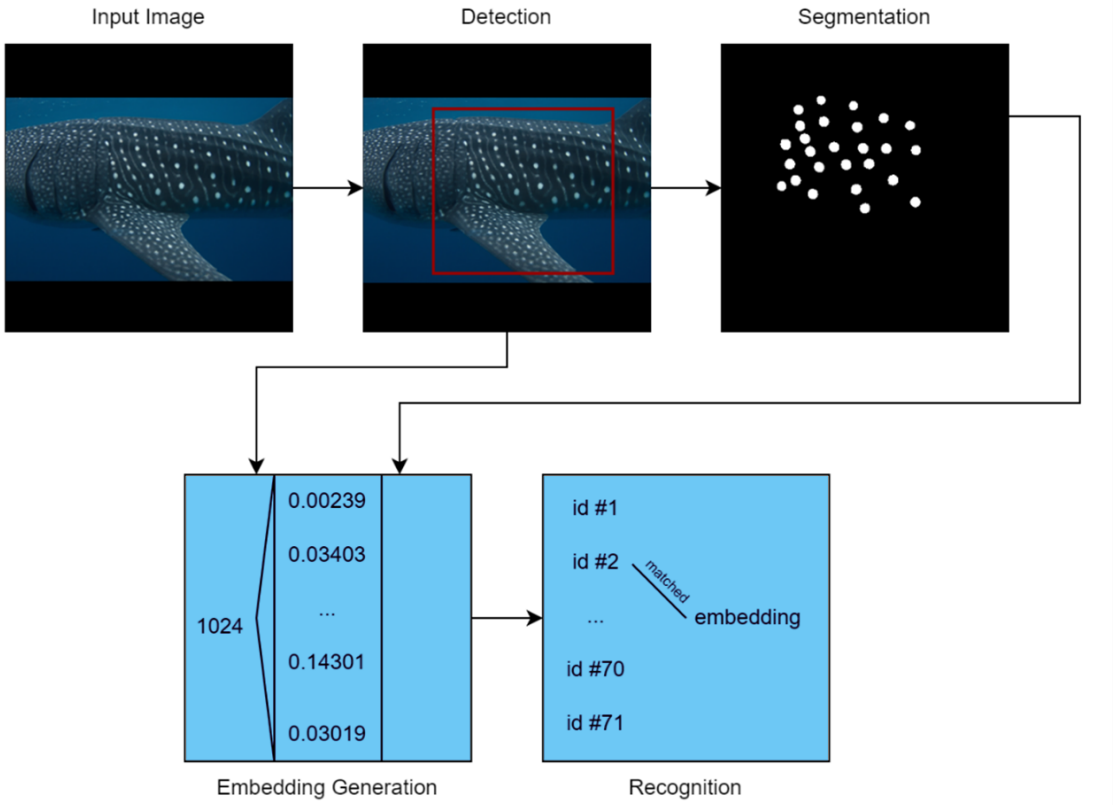

Figure 5.1: Whale shark recognition pipeline overview

At the next stage, our embedding neural network takes a detection region and segmentation mask as an input. The embedding model produces a 1024-dimensional embedding. Finally, the embedding is used in the recognition step. The recognition algorithm uses the Euclidean distance metric to compare the

embedding with the pre-computed embeddings for each individual in a database. As a result, our pipeline produces the list of the closest matches and respective distances.

5.1 Data Augmentations

We employ random crop augmentations during the deep metric learning model training. Validation and testing did not involve augmented data. The use of random crop augmentation allows the neural network to associate a broader range of spatial activation statistics with a specific class label.

5.2 Deep Metric Learning Model

Our whale shark recognition algorithm employs the Euclidean distance between embeddings as a similarity criterion. For good performance of the whale shark recognition algorithm, the Euclidean distance between instances of the same individual must be minimized, and the Euclidean distance between instances of different individuals must be maximized. To produce such embeddings, we utilize a deep convolutional neural network. Typically, such a neural network takes an image as input and produces an embedding. Our model takes as input not only an image but also a binary mask with segmented white spots on the body of a whale shark. We will show that this approach allows the model to learn additional features and generally improves recognition results. The model produces 1024-dimensional embeddings. We employ InceptionResNet [63] as the model's backbone. The InceptionResNet is a deep convolutional neural network that incorporates both Inception modules [64] and residual connections [41].

5.3 Loss Function

To improve the discriminative power of the deep features learned by the neural network, we employ the center loss function [59]. The embeddings are represented as the outputs of the penultimate layer of the neural network. The last layer represents the number of classes on which the model is trained. The loss function consists of two components. The first one is a categorical cross-entropy loss (Equation 5.1):

$$Loss_{CCE}(X,Y) = -\sum_{i=1}^{C} Y_i \log X_i \tag{5.1}$$

where X represents predicted classes and Y represents target ground-truth classes. The second one is a center loss (Equation 5.2):

$$Loss_{Center}(X,Y) = \frac{1}{2}\sum_{i=1}^{M} \left\| X_i - C_{Y_i} \right\|_2^2 \tag{5.2}$$

where X represents predicted classes, Y represents target ground-truth classes, M represents a mini-batch size, and C_{Y_i} represents a class center for Y_i. The center loss penalizes the distances between the deep characteristics of the images and their respective class centers while concurrently learning a center for each class (Equation 5.3). The update of the class center is defined as follows:

$$c_{Y_i}^{t+1} = c_{Y_i}^{t} - \alpha \Delta c_{Y_i}^{t} \tag{5.3}$$

where the hyperparameter Δ is used to control the learning rates of the centers (in our experiments, Δ= 0.95). The combined loss function is defined as follows (Equation 5.4):

$$Loss(X,Y) = Loss_{CCE}(X,Y) + \lambda Loss_{Center}(X,Y) \tag{5.4}$$

where the hyperparameter λ is used to equalize two loss functions (in our experiments, $\lambda = 0.01$).

5.4 Model Training

The model was trained on 70% of the data, 10% was used for validation, and 20% was used for testing. The model was trained for 100 epochs with a batch size of 25 and an input image size of 340 by 340 pixels. Adam optimizer with an initial learning rate of 0.05 was used. The learning rate was reduced over time using a learning rate scheduler. The model was trained on NVIDIA GeForce RTX 3080. Figure 5.2 shows the training and validation loss plot.

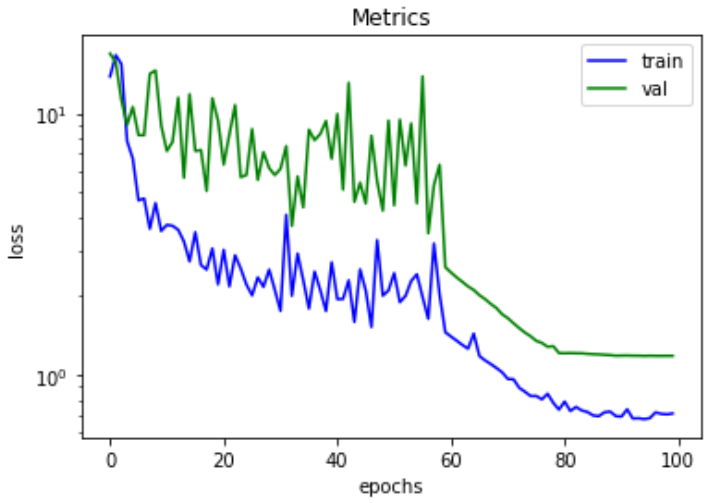

Figure 5.2: The training and validation loss plot

5.5 Recognition Algorithm

The goal of the whale shark recognition algorithm is to compare the requested embedding with the pre-computed embeddings for each individual in a database. The scheme of the algorithm is depicted in Figure 5.3.

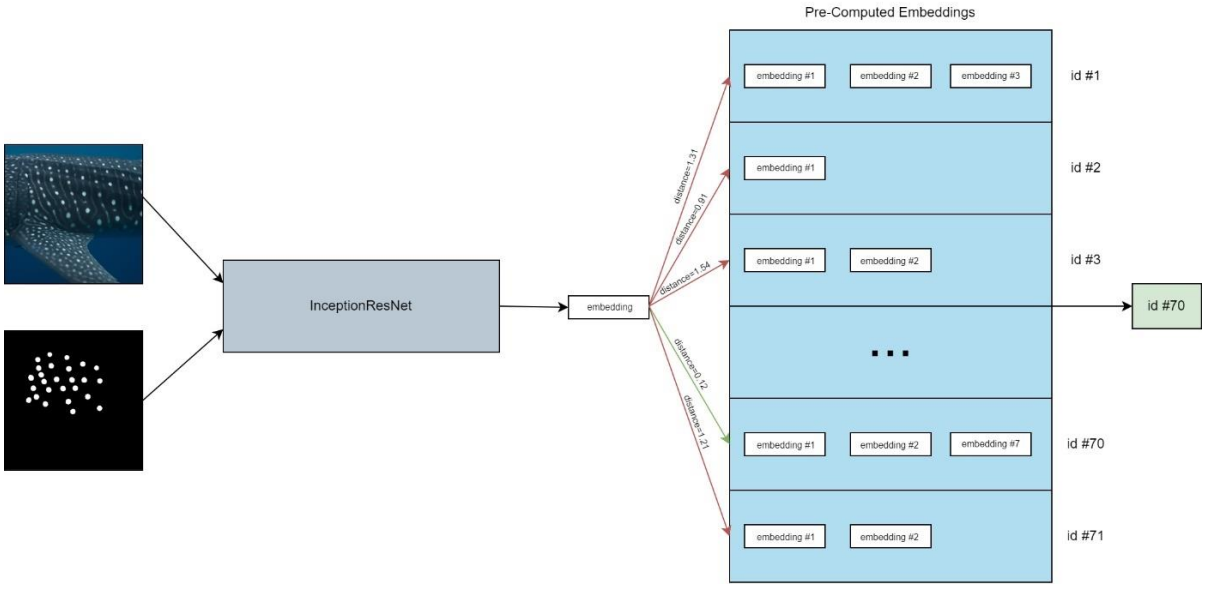

Figure 5.3: The scheme of the whale shark recognition algorithm

We use data from 71 individuals to test the recognition algorithm. For 60 individuals, we randomly select 1 image to use as a query. The remaining images are used for the recognition database. The additional

11 individuals are used to test the ability of the algorithm to detect new individuals that are not present in the database. There are a total of 48 images for these 11 individuals. Thus, we run 108 queries to evaluate the performance of the algorithm.

The algorithm computes the Euclidean distance between a query embedding and each pre-computed embedding in the database. For each individual in the database, the algorithm averages all distances to each embedding. Then, the algorithm determines the minimum average distance among all individuals in the database. If it turns out that found distance is less than the threshold (in our experiments, $threshold = 0.85$), then the query embedding belongs to the individual with the smallest distance, otherwise, the query embedding is a new individual that is not in the database.

5.6 Results

We provide performance results for the verification task and the recognition task. Verification is a problem of validating that a requested embedding matches a specific embedding (one-to-one matching). Recognition is a problem of comparing a requested embedding with all embeddings in the database (one-to-many matching).

To demonstrate the effectiveness of training a deep metric learning model on images and masks instead of using images only, we provide comparative results for verification. Table 5.1 displays the comparative results. Our approach (Images + Masks) achieves a 0.937 accuracy rate while the standard approach (Images) achieves only a 0.930 accuracy rate. The validation rate metric proposed by [50] provides a more accurate verification performance estimate. As you can see from the results below, the validation rate for our approach (Images + Masks) is 0.663 and the false accept rate is 0.0014 while the validation rate for the standard approach (Images) is 0.614 and the false accept rate is 0.0019.

Table 5.1: Performance of the InceptionResNet for verification

Input	Accuracy	Validation Rate	False Accept Rate	AUC ROC
Images	0.930	0.614	0.0019	0.976
Images + Masks	0.937	0.663	0.0014	0.982

Figure 5.4 depicts the receiver operating characteristic (ROC) curves for the model that takes images as input and the model that takes both images and masks as input. The area under the ROC curve for our approach (Images + Masks) is 0.982 while the area under the ROC curve for the standard approach (Images) is 0.976.

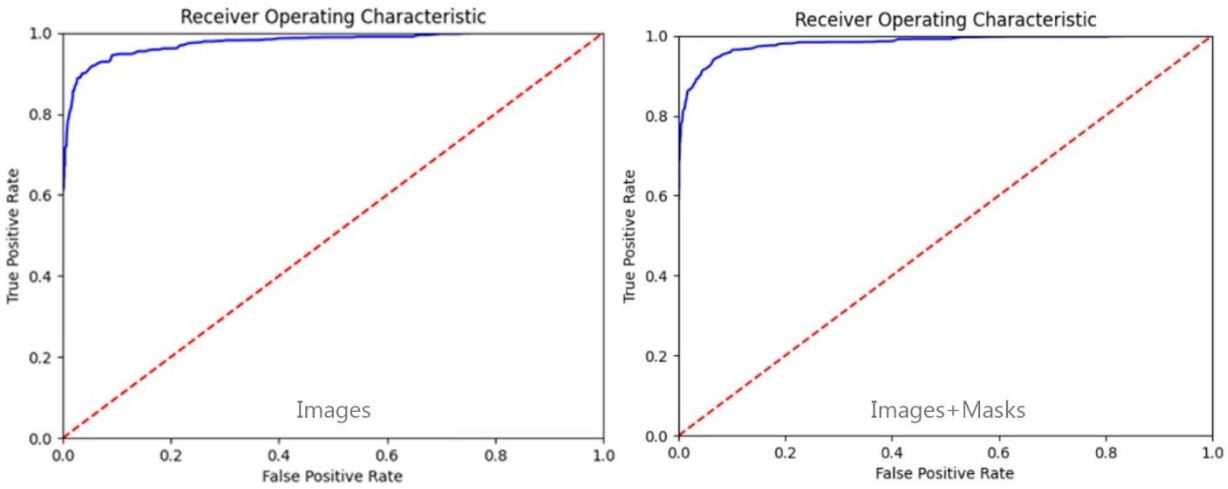

Figure 5.4: The receiver operating characteristic curves

Table 5.2 shows the performance of the recognition algorithm. We provide results for Top-1, Top-3, and Top-5 estimations. Additionally, we calculate the same metrics, but for the case when we do not need to recognize new individuals.

Table 5.2: Performance of the recognition algorithm

New Individuals	Top-1	Top-3	Top-5
No	0.93	0.98	1.0
Yes	0.83	0.96	0.98

Figure 5.5 depicts examples of correctly identified pairs of whale sharks. Green lines indicate that the distance between the embeddings is less than the threshold of 0.85. Red lines indicate the opposite. Our algorithm works pretty well with good-quality images when there are no visual artifacts and an individual is not obscured by other objects.

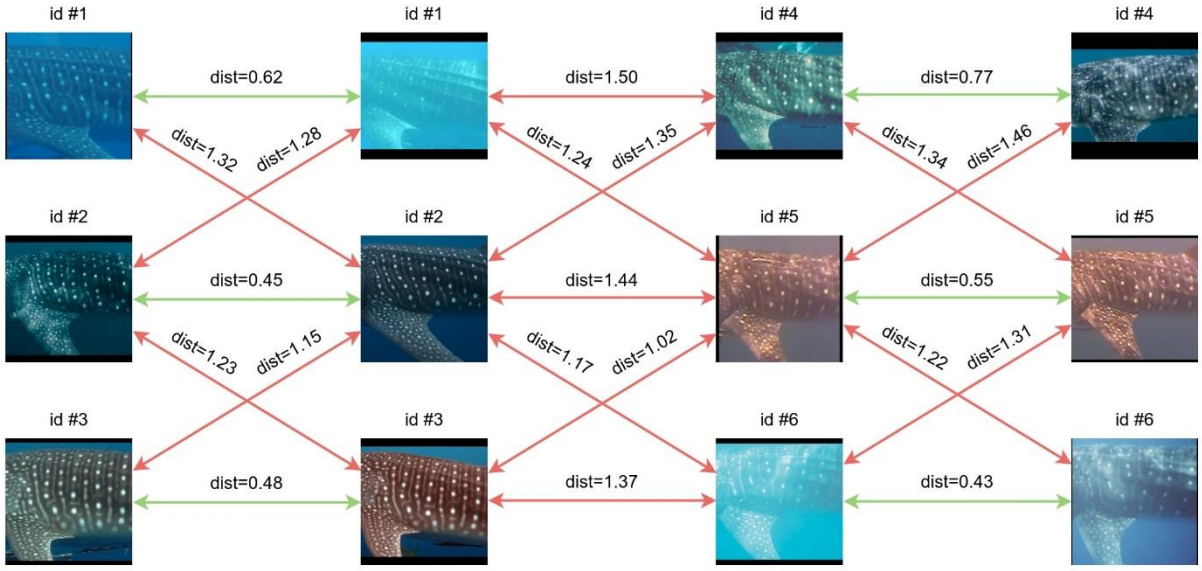

Figure 5.5: Examples of correctly identified pairs

Figure 5.6 shows examples of misidentified pairs. As we can see, in some cases of incorrect identification, the distance is still close to the threshold, but slightly exceeds it. The key factor for incorrect identification is the transfusion of light in muddy water that makes white spots on a whale shark's body less visible and darkened.

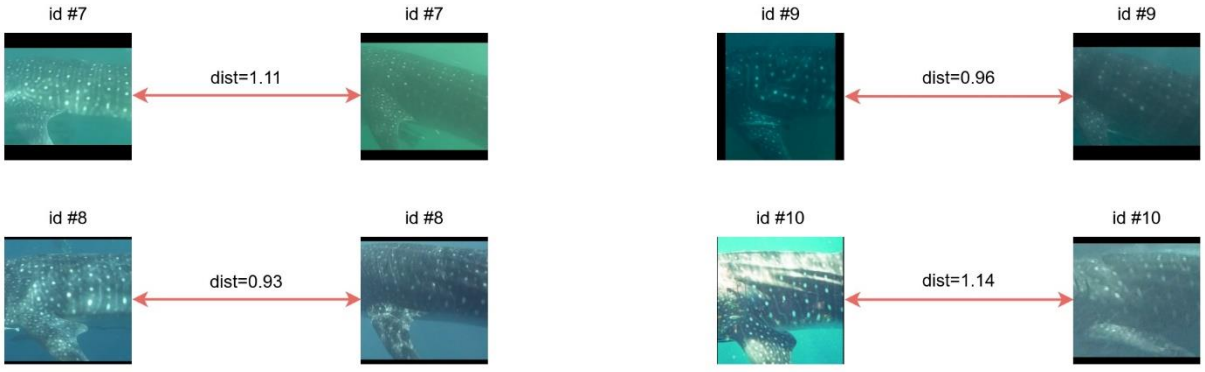

Figure 5.6: Examples of misidentified pairs

Whale sharks are usually accompanied by remoras and other small fish. They can swim alongside whale sharks and stick to them. When there are a lot of small fish around the whale shark, this can

significantly reduce the quality of recognition. Figure 5.7 depicts examples of whale sharks obscured by small fish.

Figure 5.7: Examples of whale sharks obscured by small fish

6. CONCLUSION

The conservation of the whale shark population is a pressing concern for modern society. A qualitative whale shark recognition algorithm will help solve the problem of non-invasive tracking of whale shark migration. In this work, we presented an approach for multi-stage whale shark recognition. Our algorithm enables us to work with raw whale shark image data. It automatically detects the required region and filters out photographs with inconveniently situated whale sharks or images with improper viewpoints. We employed a set of deep convolutional neural networks to create a multi-stage pipeline for whale shark recognition. Our approach consists of a region of interest detection, spot segmentation, and deep metric learning. We achieved good results for each component of our pipeline separately which allowed us to attain high recognition accuracy in general. We demonstrated that information about spot locations helps to converge embedding networks faster and leads to a better quality of embeddings. In terms of future work, we need to adapt our recognition algorithm to perform better for the data with a high amount of unseen whale shark individuals and skewed distribution of sightings. In this case, it may be necessary to develop a hybrid solution based on deep learning and an astronomical pattern-matching algorithm [16]. In addition, we plan to compare our results with the current algorithms in Wildbook.

REFERENCES

[1] A. Buetti-Dinh et al., “Deep neural networks outperform human expert’s capacity in characterizing bioleaching bacterial biofilm composition,” *Biotech. Reports*, vol. 22, no. 1, Jun. 2019, Art. no. e00321.

[2] P. Rajpurkar et al., “Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning,” 2017, *arXiv:1711.05225*.

[3] W. Zhou et al., “Ensembled deep learning model outperforms human experts in diagnosing biliary atresia from sonographic gallbladder images,” *Nat. Commun.*, vol. 12, no. 1, pp. 1–14, Feb 2021.

[4] J. Holmberg, B. Norman, and Z. Arzoumanian, “Estimating population size, structure, and residency time for whale sharks Rhincodon typus through collaborative photo-identification,” *Endang. Spec. Res.*, vol. 7, no. 1, pp. 39–53, May 2009.

[5] J. Holmberg et al., “Whildbook for Sharks.” Sharkbook. https://www.sharkbook.ai (accessed Feb. 22, 2022).

[6] W. Li, R. Zhao, T. Xiao, and X. Wang, “Deepreid: Deep filter pairing neural network for person re-identification,” *in Proc. of the IEEE Conf. on Comp. Vis. and Patter. Recognit.*, 2014, pp. 152–159.

[7] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” *in 2017 IEEE Int. Conf. on Im. Proc. (ICIP)*, 2017, pp. 3645–3649.

[8] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “Fairmot: On the fairness of detection and re-identification in multiple object tracking,” *Int. J. of Comput. Vis.*, vol. 129, no 1, pp. 3069–3087, Nov. 2021.

[9] R. Saini and N. Rana, “Comparison of various biometric methods,” *Int. J. of Adv. in Sci. and Tech.*, vol. 2, no. 1, pp. 24–30, Mar 2014.

[10] S. G. Dunbar et al., “HotSpotter: Using a computer-driven photo-id application to identify sea turtles,” *J. of Exp. Mar. Biol. and Eco.*, vol. 535, no. 1, Feb. 2021, Art. no. 151490.

[11] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. of Comp. Vis.*, vol. 60, no. 2, pp. 91–110, Jan. 2004.

[12] T. Y. Berger-Wolf et al., “Wildbook: Crowdsourcing, computer vision, and data science for conservation,” 2017, *arXiv:1710.08880*.

[13] D. Blount et al., “Flukebook.” Arabian Sea Whale Network. https://www.flukebook.org (accessed Feb. 22, 2022).

[14] T. Franklin et al., “Photo-identification of individual Southern Hemisphere humpback whales (Megaptera novaeangliae) using all available natural marks: managing the potential for misidentification,” *J. Cetacean Res. Manage.*, vol. 21, no. 1, pp. 71–83, Oct. 2020.

[15] J. P. Crall, C. V. Stewart, T. Y. Berger-Wolf, D. I. Rubenstein, and S. R. Sundaresan, “Hotspotter—patterned species instance recognition,” *in 2013 IEEE Workshop on Appl. of Comp. Vis. (WACV)*, 2013, pp. 230–237.

[16] Z. Arzoumanian, J. Holmberg, and B. Norman, “An astronomical pattern-matching algorithm for computer-aided identification of whale sharks Rhincodon typus,” *J. of Appl. Ecol.*, vol. 42, no. 6, pp. 999–1011, Nov. 2005.

[17] E. J. Groth, “A pattern-matching algorithm for two-dimensional coordinate lists,” *The Astron. J.*, vol 91, no. 1, pp. 1244–1248, May 1986.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2014, pp. 580–587.

[19] R. Girshick, “Fast r-cnn,” *in Proc. of the IEEE Int. Conf. on Comp. Vis.*, 2015, pp. 1440–1448.

[20] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Trans. on Pat. Anal. and Mach. Intel.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[21] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2018, pp. 6154–6162.

[22] G. Jocher et al. *ultralytics/yolov5* (2022). Zenodo. Accessed: Feb 22, 2022. [Online]. Available: https://github.com/ultralytics/yolov5

[23] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2016, pp. 779–788.

[24] J. Redmon and A. Farhadi, “YOLO9000: better, faster, stronger,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2017, pp. 7263–7271.

[25] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” 2018, *arXiv:1804.02767*.

[26] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” 2020, *arXiv:2004.10934*.

[27] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, “You only learn one representation: Unified network for multiple tasks,” 2021, *arXiv:2105.04206*.

[28] M. Tan, R. Pang, and Q. V. Le, “Efficientdet: Scalable and efficient object detection,” *in Proc. of the IEEE/CVF Conf. on Comp. Vis. and Pat. Recognit.*, 2020, pp. 10781–10790.

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” *in Proc. of the IEEE Int. Conf. on Comp. Vis.*, 2017, pp. 2980–2988.

[30] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, “Centernet: Keypoint triplets for object detection,” *in Proc. of the IEEE/CVF Int. Conf. on Comp. Vis.*, 2019, pp. 6569–6578.

[31] J. Parham, C. Stewart, J. Crall, D. Rubenstein, J. Holmberg, and T. Berger-Wolf, “An animal detection pipeline for identification,” *in 2018 IEEE Wint. Conf. on Appl. of Comp. Vis. (WACV)*, 2018, pp. 1075–1083.

[32] Q. Sellat, S. Bisoy, R. Priyadarshini, A. Vidyarthi, S. Kautish, and R. K. Barik, “Intelligent semantic segmentation for self-driving vehicles using deep learning,” *Comput. Intel. and Neurosci.*, vol. 2022, pp. 1–10, Jan. 2022.

[33] X. Liu et al., “Importance-aware semantic segmentation in self-driving with discrete wasserstein training,” *in Proc. of the AAAI Conf. on Artif. Intel.*, vol 34, 2020, pp. 11629–11636.

[34] S.-H. Zhang, X. Dong, H. Li, R. Li, and Y.-L. Yang, “PortraitNet: Real-time portrait segmentation network for mobile device,” *Comp. & Graph.*, vol. 80, no. 1, pp. 104–113, May 2019.

[35] X. Nie, M. Duan, H. Ding, B. Hu, and E. K. Wong, “Attention mask R-CNN for ship detection and segmentation from remote sensing images,” *IEEE Access*, vol. 8, pp. 9325–9334, Jan. 2020.

[36] R. Ranjbarzadeh et al., “Lung infection segmentation for COVID-19 pneumonia based on a cascade convolutional network from CT images,” *BioMed Res. Int.*, vol. 2021, pp. 1–16, Apr. 2021.

[37] F. Gholamiankhah et al., “Automated lung segmentation from CT images of normal and COVID-19 pneumonia patients,” 2021, *arXiv:2104.02042*.

[38] F. Milletari, N. Navab, and S.-A. Ahmadi, “V-net: Fully convolutional neural networks for volumetric medical image segmentation,” *in 2016 Four. Int. Conf. on 3D Vis. (3DV)*, 2016, pp. 565–571.

[39] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *in Int. Conf. on Med. Im. Comp. and Comp.-Assist. Interv.*, 2015, pp. 234–241.

[40] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, *arXiv:1409.1556*.

[41] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2016, pp. 770–778.

[42] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” *in IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2018, pp. 7132-7141.

[43] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2017, pp. 1492–1500.

[44] A. Chaurasia and E. Culurciello, “Linknet: Exploiting encoder representations for efficient semantic segmentation,” *in 2017 IEEE Vis. Commun. and Im. Proc. (VCIP)*, 2017, pp. 1–4.

[45] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” 2017, *arXiv:1706.05587*.

[46] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” 2015, *arXiv:1511.07122*.

[47] A. G. Howard et al., “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” 2017, *arXiv:1704.04861*.

[48] I. T. Jolliffe and J. Cadima, “Principal component analysis: a review and recent developments,” *Phil. Trans. of the Roy. Soc. A: Math., Phys. and Eng. Sci.*, vol. 374, no. 2065, Apr. 2016, Art. no. 20150202.

[49] L. Van der Maaten and G. Hinton, “Visualizing data using t-SNE,” *J. of Mach. Learn. Res.*, vol. 9, no 86, pp. 2579–2605, Nov. 2008.

[50] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2015, pp. 815–823.

[51] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” *in Proc. of the IEEE/CVF Conf. on Comp. Vis. and Pat. Recognit.*, 2019, pp. 4690–4699.

[52] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz, “Joint discriminative and generative learning for person re-identification,” *in Proc. of the IEEE/CVF Conf. on Comp. Vis. and Pat. Recognit.*, 2019, pp. 2138–2147.

[53] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” 2017, *arXiv:1703.07737*.

[54] Z. Zheng, L. Zheng, and Y. Yang, “A discriminatively learned cnn embedding for person reidentification,” *ACM Trans. on Mult. Comp., Commun., and Appl. (TOMM)*, vol. 14, no. 1, pp. 1–20, Nov. 2016.

[55] Z. Zheng, T. Ruan, Y. Wei, Y. Yang, and T. Mei, “VehicleNet: Learning robust visual representation for vehicle re-identification,” *IEEE Trans. on Mult.*, vol. 23, pp. 2683–2693, Apr. 2020.

[56] E. Pianin, “Image segmentation and deep metric learning for whale shark re-identification,” M.S. thesis, Dept. Comp. Sci., Rensselaer Polytechnic Inst., Troy, NY, 2020.

[57] L. Bergamini et al., “Multi-views embedding for cattle re-identification,” *in 2018 14th Int. Conf. on Sign.-Im. Technol. & Intern.-Based Sys. (SITIS)*, 2018, pp. 184–191.

[58] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” *in 2005 IEEE Comp. Soc. Conf. on Comp. Vis. and Pat. Recognit. (CVPR 2005)*, vol. 1, 2005, pp. 539–546.

[59] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, “A discriminative feature learning approach for deep face recognition,” *in Europ. Conf. on Comp. Vis.*, 2016, pp. 499–515.

[60] J. S. Bridle, “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition,” *Neurocomp.*, vol. 68, no. 1, pp. 227–236, Jan. 1990.

[61] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, “Path aggregation network for instance segmentation,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2018, pp. 8759–8768.

[62] S. Jadon, “A survey of loss functions for semantic segmentation,” *in 2020 IEEE Conf. on Comput. Intell. in Bioinform. and Comput. Biol. (CIBCB)*, 2020, pp. 1–7.

[63] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” *in Thir.-first AAAI Conf. on Artif. Intell.*, 2017, pp. 4278–4284.

[64] C. Szegedy et al., “Going deeper with convolutions,” *in Proc. of the IEEE Conf. on Comp. Vis. and Pat. Recognit.*, 2015, pp. 1–9.