Whale Identification Based on Siamese Neural Network

Guang Yang[1]

Abstract— **Since the industrial whaling in the 17th century, more than one million whales have been killed in worldwide [1]. This whaling scale has affected most whale populations, and it has significantly altered their ecological role in the marine environment. Several species of the whales have already been listed in the IUCN's (International Union for Conservation of Nature) Red List of Endangered Species with fragile, endangered or extremely endangered. Some species were classified as data scarcity, which means that the remaining amount of these species is not enough to be evaluated, and it increases the difficulty in protecting whales.**

To protect the whales, scientists used photo surveillance systems to monitor marine activities. They use the shape of the whale's tail and the unique markers in the lens to identify the type of whale. They are analyzing and carefully record the dynamics and movement of the whale pod. In the past 40 years, scientists have worked hard to record and protect, and they have left large amount of valuable untapped and underutilized data. My project is in the the Kaggle Kernel https://www.kaggle.com/codegass/siamese-net-with-ensemble And the final score of the model is 0.93808 on private leaderboard and 0.93388 on public leaderboard. The Ranking is up to top 4%.

I. Introduction

In this project, I used the 25,000 images of the tails (fluke) provided by happywhale.com on the Kaggle competition, and I tried to build the model to maximize the accuracy of the whale identification. In this process, I modified the image data processing, used the bounding box model, Siamese Neural Network, and ensemble method as the model structure . The report present my experience to train the model, the concept for each step, and the solutions when I had issues.

The approach that I used in this project is essentually a SNN (Siamese Neural Network), with some modifications that will be described in the report later. For preparing this project, I read some relative papers about Siamese Neural Network [2], Triplet Loss [3], and Squeeze-and-Excitation Networks [4]. I also studied and discussed other competitors' concepts from their Kernals and blogs. Martin Piotte's kernel is my main learning and reference. Many of my concepts and solution are also based on the improvement of his ideas.

II. Relative Works

There have been previous applications of computer vision to recognition of individuals (e.g., Arzoumanian et al. 2005; Crall et al. 2013; Beijbom et al. 2015), including whales (e.g., Hiby Lovell 2001; Kniest 2010; Flukebook 2018). For most of these approaches start by extracting certain features of the target object from a huge number of images to train the machine learning model.

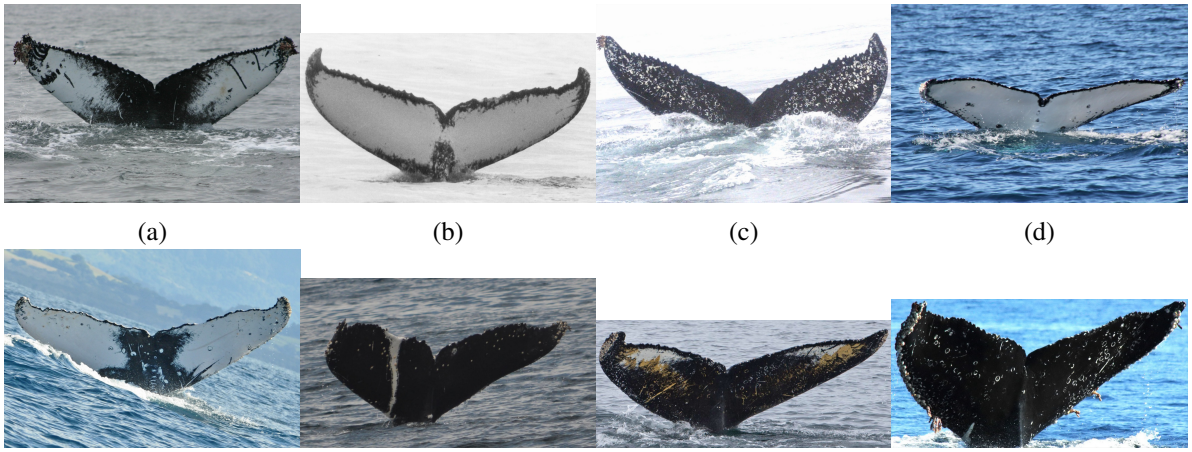
These features are typically obtained by applying standard filters to the input image (e.g., Gabor filters, DoG filters, etc.) with the choice of filters based on experimentation or experience from manual individual recognition.

III. Dataset Analysts and Process

A. Image Data Analysts

For this competition, there were totally over 25,000 images for training, and around 8,000 images for testing. The main goal for us was to identify the single image of the whale tails that belonged to the 5,000 known whales or belonged to the unknown class new_whale. Most of data are normal vertical images, and each image has relatively high resolution in the original format. (Fig. 1) However, not all the data are good to train the model. There are some images exist the problems. I need to classify and modify those images.

1) Unknown and Unbalanced Issues: The dataset has unknown and unbalanced issue. The distribution of images per whale (IPW) is highly skewed. There were more than 2,000 whales only have one image, and the single whale that had the most images had 73 images. In the dataset, about



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 1: Sample Images

30% whales IPW are equal or less than 4; about 40% whales come from new_whale class; about 30% whales IPW are between 5 to 73 images. (Shows in Fig.2) Unknown issues: Out of 25,000 whales images, there were total 5,005 classes. More than 9,600 images of the whales belong to the new_ whale class, which means unidentified whales. However, I cannot simply group those images as one class for training because some of them belonged to on category, but I do not know them and cannot directly use them for training.(Shows in Fig.3)

I do not know how to fix the issue about new_whale class. Therefore, I used the easiest and safety method that remove them from the training data. After removing data, it left about 15,000 effective training images, which belonged to the 5,004 classes in total. However, I still faced to the other challenge, unbalanced issue. The contribution of the dataset was very skewed and unbalanced. More than 2,000 classes only had one images, more than 1,250 classes only had two images. The traditional machine learning methods were not very useful for the skewed dataset. I wanted to fix the dataset issue, I selected to use SNN (Siamese Neural Network) to adjust the dataset.

2) The Resolution Issue: Most of images are clearly and have the good resolution such as 1050*700 dimension. However, there were some bad-resolution images, which the images were distorted or not displayed properly (Showed in Fig.

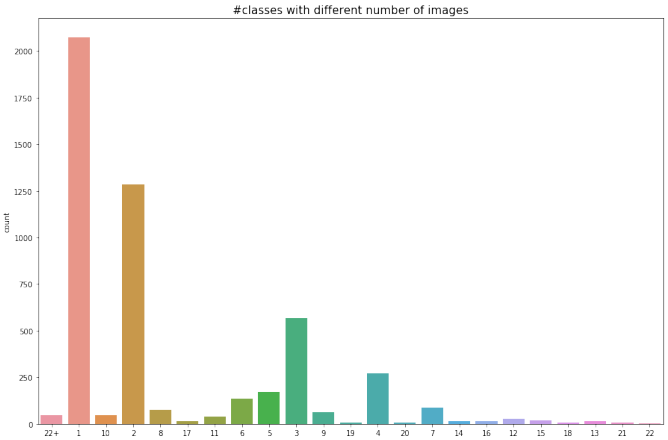


Fig. 2: The Number of each amount of whale's image numbers Tagged Whale

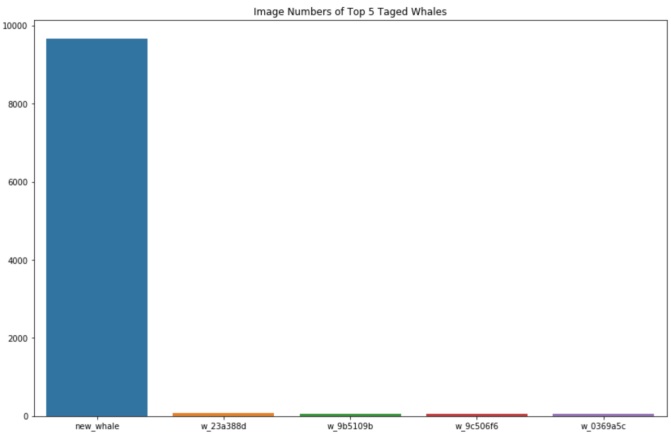


Fig. 3: The Image Numbers of Top 5 Tagged Whale

4). The problematic images had some impacts for the model, but only minor images were like those. Therefore, I just ignored the impact of the images and assumed they would not have big impact on the models.

B. Image Processing

In image data analysts, I mentioned that images like in new_class that I cannot handled well and use in my training data, so I removed those images. Also, like bad-resolution and distorted images, because there are is only a small amount image and it will not affect my results, we will keep them. However, improving the input images can still help us better to train my model. In data processing, I continued to modify the image inputs. I used duplicate image identification and bounding box.

1) Duplicate Image Identification: Before building the model, I processed the training images again because I found some repeatable images. First, they cannot improve my training result but cause over-fitting. In addition, they would increase the training time. Also, they will increase the inequality of the data set. So, I need to identify similar images and remove the extra one. I defined that the distance between any two normalized images is less than 0.1. The images are similar and repeated.

2) Bounding Box: I realized that the bounding box could help us solve the problems of the remaining data when I saw the Martin's forum in the discussion. While many of the whale images in the dataset are already cropped tight around the whale fluke, in some images the whale fluke occupies only a small area of the image. So I use the bounding box to zoom in images with keep the important features and removed the unnecessary parts such as the ocean. I use CNN (Convolution Neural Network) to implement this process, and at the same time, help us solve the problem of images rotation. Using this model, whale images were cropped automatically to a more uniform appearance. It facilitated training of classification models and improved the test accuracy. I directly used Martin's weights in this part, the size in final cropped was 384 by 384.

IV. Model Structure: (SNN) Siamese Neural Network

When I see image problems, I tried to use the traditional Convolution Neural Network first. (Fig. 5) However, the result was not satisfied, only with accuracy 0.327. In this project, there were many categories, but the number of samples in each category is small. There is not enough training data can be used for category identification and classification. Since there are too few samples in each category, I cannot train the good results at all. Siamese Neural Network could help us solve this issue.

A. Introduction of Siamese Neural Network

In the early 1990s, Bromley and Lecun first use Siamese nets to solve signature verification as an image matching problem [3]. The Siamese Neural Network based on the Convolution Neural Network (Fig. 6) to extract the specific features from the two input images, and use Sigmoid layer as the output layer to determine whether two images belong to the same category or not. The function computes some metric between the highest-level feature representation on each side.

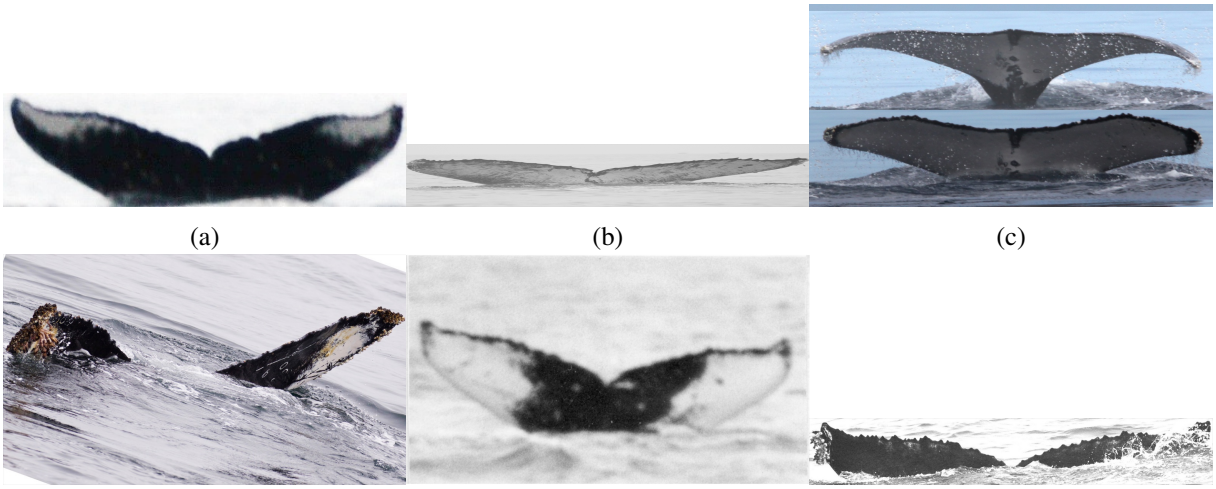
During training, the inputs of images are always paired because they will be inputted two same convolutional layers. The feature vectors produced after they crossed the convolutional layers. The units in the final convolutional layer are flattened into a single vector [2]. A fully-connected layer follows the convolution layer. Then, using one more layer calculates the metrics distance between each Siamese twin, which is given to a sigmoidal output:

$$P = \delta_i(\sum_i \alpha|h_1^i - h_2^i|)$$

where σ is a sigmoidal activation function

B. Branch Model

The branch model is a regular Convolution Neural Network model. For the branch model selected, I did the different attempts. For the first attempt, I used self-design Convolution Neural Network



(a) (b) (c)

(d) (e) (f)

Fig. 4: Bad Resolution and Distorted Images

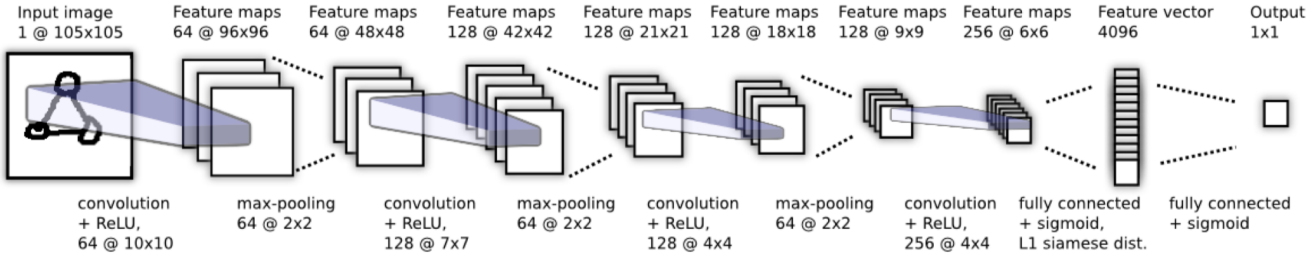


Fig. 5: Best convolutional architecture selected for verification task. Siamese twin is not depicted, but joins immediately after the 4096 unit fully-connected layer where the L1 component-wise distance between vectors is computed

layer, which is composed of 6 convectional layers:
Layer 1 - 384*384
Layer 2 - 96 * 96
Layer 3 - 48 * 48
Layer 4 - 24 * 24
Layer 5 - 12 * 12
Layer 6 - 6 * 6
For the second attempt, I attempted pre-trained ResNet.
For the third attempt, I attempted pre-trained DenseNet.
With local cross-validation test, the DenseNet121 has the best performance. However, it can only work on my local desktop with strong GPU support, not on Kaggle kernel. The Kaggle kernel has limited 9 hours each session, and it cannot finish the training in time. I used the different type of branch models and also produced the different hyper parameters, so I used all the result to ensemble the final submission.

C. Head Model

After I got the feature vectors from the branch model, I want to use head model to determine whether inputted the twin whales were the same. The most traditional method is to use a distance measure the loss function. However, there were some limitations in this project.

First, I know it is perfect match if the values of both two features were zero. Even both features with a large value, but they only have very sightly different, I still assume they have a good match. In other case, there are two images have same future X, they must be the same image. However, feature X is not as clear if both images also had feature Y. Also, if two images X, Y were same each other, swapping two images will not change any results. Based on those concepts, I did the following step:

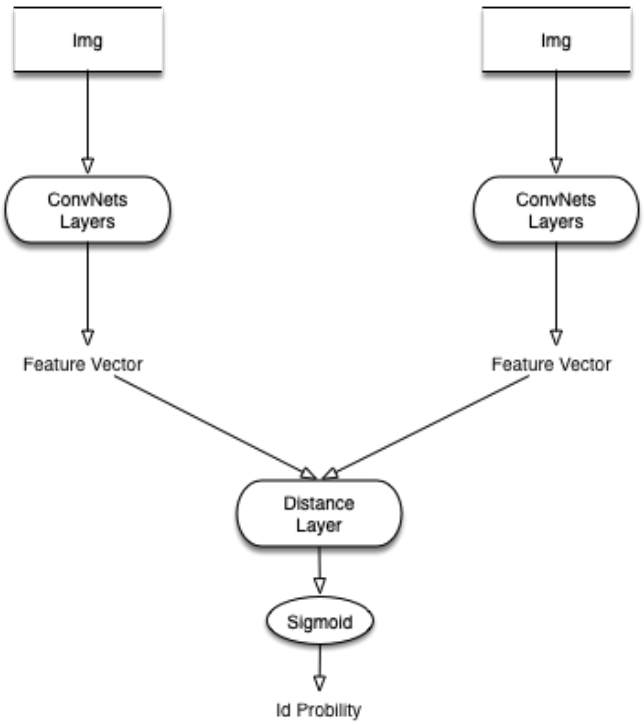


Fig. 6: Siamese Neural Network (SNN) Structure

Step 1: computing the sum, product, absolute difference, and squared difference:

$$[x + y, x, |x - y|, (x - y)^2]$$

Step 2: I built a small neural network for learning how to weight between zeros-value features and non-zero-value features.

I just saw the other concept in this part that uses triple model to find the distance. I considered it as well, and put my concepts in Future Work paragraph.

V. Training and Ensemble

Training the large model from random weights is difficult. In this context, with the random initial weight for the training model will cost nearly 60 hours to converge. And because of the large scales of the data, there will always be some cases that two different whale's pictures looked far more same than the same whale's two different pictures. To avoid these hard cases problem, the model uses a constant K measuring the scale of the random component of the score matrix used to match similar images to construct tough training cases in each epoch. As the model ability to distinguish between whales increases, K is gradually reduced, presenting harder training cases.

To boosting the training progress, I also do the warm training start. I use the pre-trained model weights as initialization for CNN layers and I design a strategy let the beginning of the training is different from the later epochs. This means at the beginning of the training I don't give the model a L2 regularization, but after 250 epochs , which in this case means the model nearly have ability to identify most of the whales but also grossly to over-fitting, I add the L2 regularization to the model and set learning rate to a large value for the next hundreds of epochs.

I run multiple experiments with different hyper parameters and using different CNN layers for the branch model in SNN, so I have different results to do ensemble. Here I use the weighted average ensemble learning method and use the data from best 4 different cases to build the final result. For submission data which has higher score I give it higher weight.

Consider of the ability of the kaggle kernel, I trained most of the model on local computer and input the weights as pre-train when running the rest of epochs in kaggle kernel.

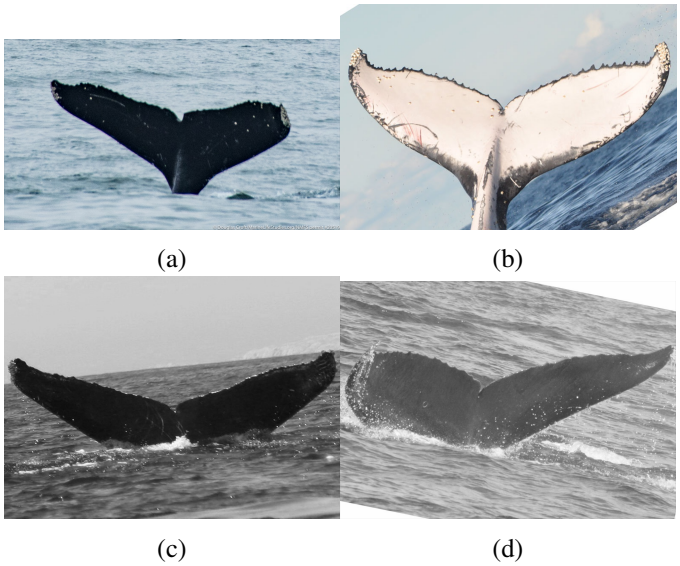
VI. Future Work

Although I have finished the project with a good result, I still have some concepts did not to attempt due to the time limit. I do not know whether those attempts are able to improve my result, but I still want to try those concepts. I create a future work list to shows what I want to try:

- Adversarial Training Procedures
- Boots

A. Adversarial Training Procedures

Adversarial training procedure (Fig. 7) is proposed by other competitors. They discovered the dataset existed easier cases and hard cases after they repeatably trained. If they started to use hard samples to feed the model, the model might predict the similar images as the different whales and predict the dissimilar images as same whales. It reduces the accuracy of the model, and the training fails eventually. Therefore, they started to feed the easier samples by randomly sampling matching and un-matching pairs. They started to



(a) (b)

(c) (d)

Fig. 7: Easy Case vs. Hard Case
Figures (a) and (b) are the easy cases, Figures (c) and (c) are the hard cases

feed the hard samples after the training process and model becomes stronger. Since the only same-class whales are matched pairs, and only different-class whales are un-matched pairs, the un-matched pairs are much more than matched pairs. They only form adversarial pairs from the un-matched whales.

I think the method can improve the performance of the model, and I will attempt to use and improve the method in future work.

B. Bootstrapping

The Bootstrapping algorithm refers to the re-establishment of a new sample sufficient to represent the distribution of the parent sample through repeated sampling using a limited sample of data. The use of bootstrapping is based on many statistical assumptions, so the accuracy of the sampling will affect the establishment of the hypothesis.

Some competitors used Bootstrapping, but most of them gave a negative feedback. I do not know whether it could improve my model or also failed, but I still want to try this method in future.

REFERENCES

[1] Whale protection, Department of the Environment and Energy, May 19, 2019, URL: https://www.environment.gov.au/marine/publications/factsheet-whale-protection

[2] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, Siamese Neural Networks for One-shot Image Recognition, 2015.

[3] Florian Schroff, Dmitry Kalenichenko, James Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering, 17 June, 2015.

[4] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, Enhua Wu, Squeeze-and-Excitation Networks, 2019.

[5] Martin Piotte, 2018, Whale Recognition Model with Score 0.78563, URL: https://www.kaggle.com/martinpiotte/whale-recognition-model-with-score-0-78563