Machine Learning Engineer Nanodegree

Capstone Project

Ruba Rammal

March 19th, 2018

I. Definition

Project Overview

The Right Whales are species of whales that are threatened with extinction, so in order to protect the whales and monitor their health, researchers track their locations by taking photographs during aerial surveys. These photos are then matched by scientists against an existing catalog to determine the species by the aid of a software.

The system does not automate the process and depends on manual inspection which is time consuming and assumes that the researcher have knowledge in matching the whales so MathWorks sponsored a Kaggle[1] competition to automate this process using the whale images dataset labeled by Christin Khan and Leah Crowe from NOAA and the photo-identification catalog provided by New England Aquarium.

Some of the most common methods for solving image classification problems are supervised learning techniques. By applying image segmentation and extracting features and feeding them to a supervised learning algorithm like SVM or Random Forest or a an Ensemble of algorithms[2]. Then Deep Learning and Convolutional Neural Networks made it possible to further understand images and detect or recognize certain shapes and objects and make reliable predictions[3]. One of the most important work done in that field is the ImageNet neural network where a network was trained to classify 1.2 million images with high resolution[4]. CNNs are suitable for automating the whale recognition process for they have proved in previous research to make accurate predictions with image classification problems.

Problem Statement

To automate the process of identifying right whales which is an image classification problem, the labeled dataset that contains 447 unique whale ids will be used to build a whale face recognition model that can detect and classify right whales which will make the process of targeting the whale to monitor its health and help save it from extinction much more efficient.

Using the right whale images dataset, a convolutional neural network will be trained to classify an input image as one of the 447 classes. The model will take one image as an input and will output 447 probabilities of how likely does the image belong to each class.

Metrics

The log loss (or loss-entropy)[5] formula is used as a loss function when training the model, log loss produces a prediction values between 0 and 1.

$$Logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

The goal is to minimize this value so the closer the log loss value is to zero the better the prediction is.

Log loss gives more nuanced measure of performance[6]. It gives high penalty to wrong classifiers so for example if the value is 1 and the classifier predicts that it is 0.1, the value of the log loss will be very high[7]. This formula is suitable for image classification problems because it measures how much a predicted classification is close to the label instead of providing a yes no evaluation.

II. Analysis

Data Exploration

The dataset compromises of different aerial images of 447 unique whales, the pictures were taken over the course of ten years using different equipment so the color of each image differs based on the time and equipment used to capture the photo, and each image contains a single whale. The images were picked and labeled by scientists at NOAA.

Each whale has a unique mark on its head that distinguishes it from other whales and this mark is not very clear in every image and that creates the challenge of identifying the whale. The images can be cropped prior to training the model to focus only on the whale and remove the sea water areas in the image.

FIG. 1 SAMPLE IMAGES OF WHALES

The training set provided in the competition contains 4544 images and the whole set consists of 11469 images, however, 6925 images are not labeled since Kaggle evaluates the model after submission so the training images will be used for training and testing in this case.

The data set contains 4544 images that are not well distributed, some whales have more images than others and many of the whales have only one image. The images are JPEG format, 2D and composed of three arrays for different color channels (RGB) but they vary in brightness and color and they range in dimensions as well since different equipment were used over ten years to capture them. Some of the largest images is 5184 x 3456. All of these variations as well as the splashes of water that can be misclassified as head marks create a challenge when attempting to build a robust classifier to identify an individual whale.

Exploratory Visualization

Figure 2 visualizes the number of occurrences of 30% of the whales in the image set.

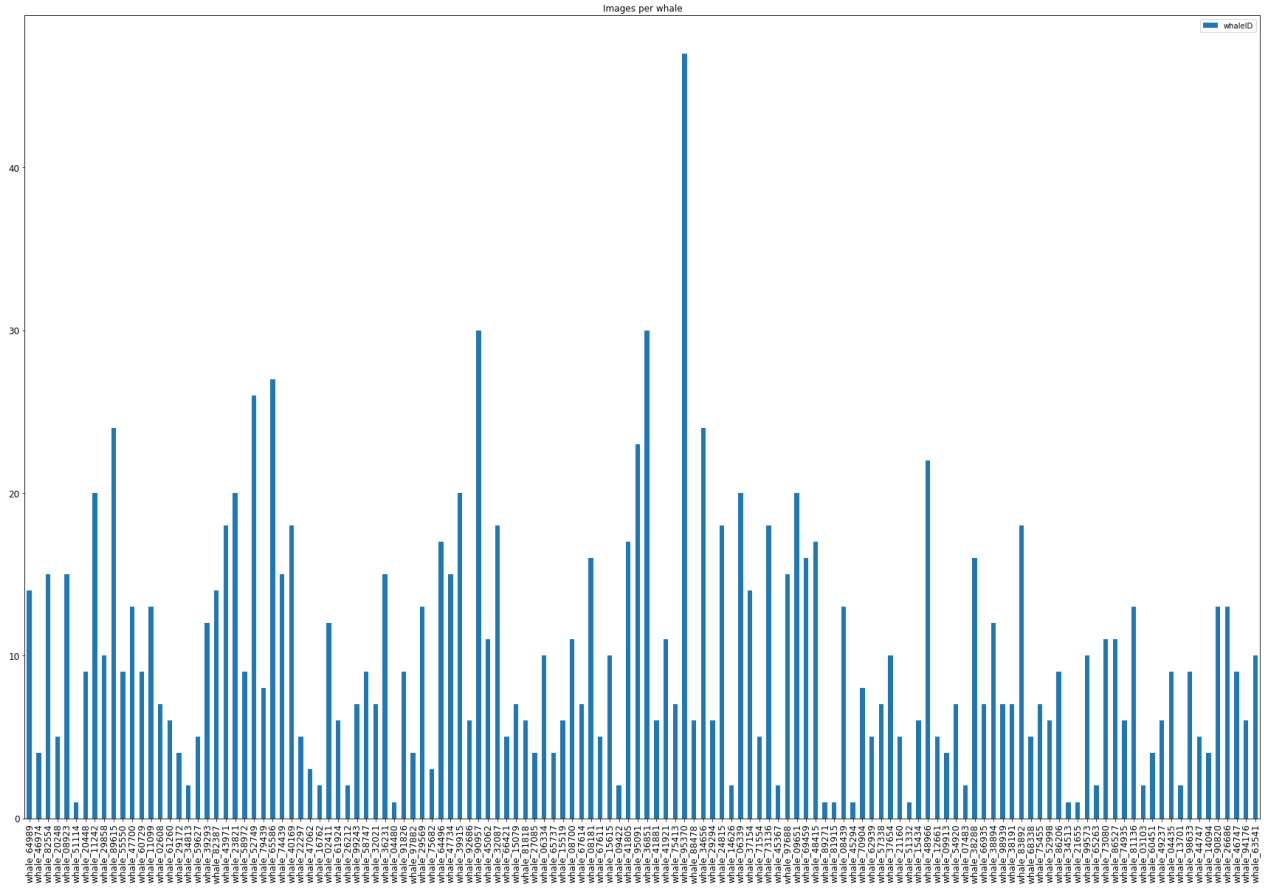

FIG. 2 IMAGES PER WHALE

This shows how imbalanced the dataset is, some whales have fifty images and some have one or two images only which will make training the model to correctly classify all 447 whales a difficult task. This shows that data augmentation on the whaleIDs that have small numbers of images might be needed to attempt balancing out the images and expanding the training set prior to training.

Another thing to note about the images is how the only relevant part is the head to tail part of the whale (most importantly the head mark), most images as can be seen in figure 3 contain large areas of sea water. Cropping the images to focus only on the whales' heads will make a difference in training.

Algorithms and Techniques

A Convolutional Neural Network is the classifier that will be developed to solve this problem since deep learning has proven to be the best method for computer vision and image classification problems. The way that a CNN classifier works is by extracting features from the image and then for every layer, a filter detects the presence of certain features. The deeper you go into the network, the less general the predicted features are and the final layer usually combines all of the layers' predictions to produce the final output[8]. While training, the weights of each layer are updated in the backpropagation process to attempt minimizing the error produced when comparing the prediction to the image's label.

The following are the ways which can improve the model's performance[9]:

1. Data preprocessing (see Data Preprocessing section)
2. Model architecture
   (a) Layers: the number of layers and their different types (convolutional, pooling, fully connected, dropout).
   (b) Layer parameters: filters, kernel size, activation functions (ReLU and Softmax)
   (c) Transfer learning: used to obtain good accuracy with shorter or no training time by using pre-trained architectures like VGG16 or VGG19, freezing some layers and training the remaining layers and any added custom layers on the dataset.
3. Algorithm parameters
   (a) Epochs: the number of times the model will train on the images.
   (b) Batch size: the number of images the model will train with at once (load into memory).

Benchmark

A CNN with 2DConv layer, global average pooling layer and a fully connected layer was trained using the preprocessed dataset to obtain a benchmark and resulted in an accuracy of 1% (5.99 loss rate).

III. Methodology

Data Preprocessing

The following steps were applied to preprocess the data prior to training the model:

1. One hot encoding was applied to the labels (whaleIDs) using scikit-learn preprocessing LabelEncoder and OneHotEncoder.
2. Applied processing on the images that focuses on the whales' head to tail areas by checking the differences of the colors in the images hisotogram and crops the image accordingly, the code was taken from Eduardo Flores' Github[10].
   But since tensor requires preprocessing the images and scaling them down, the images must have the same shape so the function was modified to check the largest dimension and to set it to both height and width. (see Figure 3)
3. Converted images to 4D arrays (samples, columns, rows, channels) to be compatible with TenserFlow requirements.
4. The 4D arrays were divided by 255 to rescale the images.
5. Applying PCA on the features was considered, but Felix Lau the second place winner of the competition[11] tried it in his implementation and reported that it did not improve the accuracy so feature selection was not implemented.

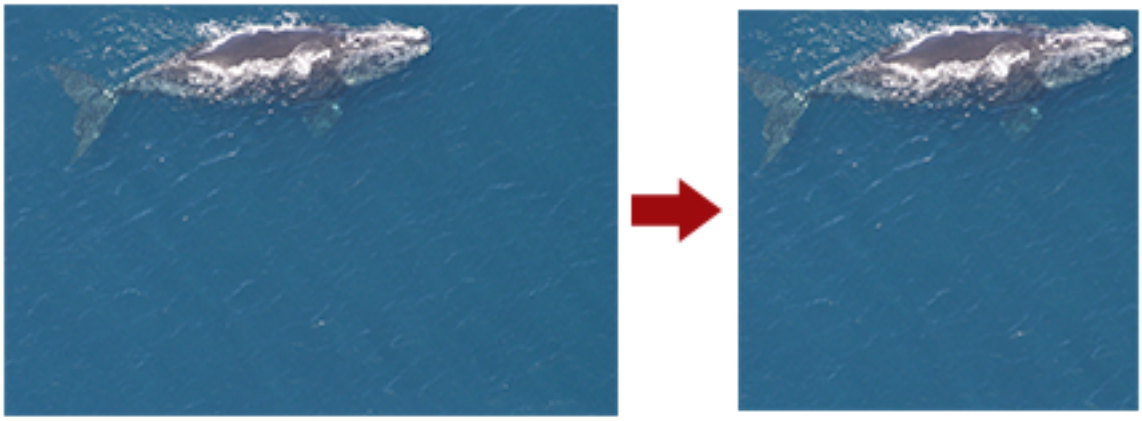

FIG 3 BEFORE AND AFTER CROPPING THE IMAGE

Implementation

The following steps summarize the implementation of the model:

1. Load the data.
2. Apply data preprocessing (see Data Preprocessing section).
3. Split the dataset into training and testing sets using scikit learn's train_test_split.
4. Apply Keras preprocessing (reshape, rescale).
5. Build the benchmark model and train it with the full training set.
6. Implement the model architecture:

(a) The first solution that was attempted included building the model by using VGG19 as a base model and setting the initial weights to the imagenet weights, freezing the first five layers and adding two more custom layer (see figure 4). This model trained for almost 30 hours but it overfitted and did terribly on the testing set (11.5 cross entropy rate).
(b) The second solution was a CNN architecture that corresponds to the original proposed solution (see figure 5) but it was slightly modified to include a Dropout layer to avoid overfitting.
(c) The input shape is (width, height, channels) which corresponds to (256, 256, 3) in this case.
(d) Set the model's loss or optimization function to the categorical cross entropy and compile the model.
(e) Set the training parameters and train the model.

One of the issues faced was related to image augmentation. There is a certain folder structure that Keras requires to apply augmentation and it did not match the existing structure of the dataset so only real-time image augmentation was possible in this case which did not work well on training.

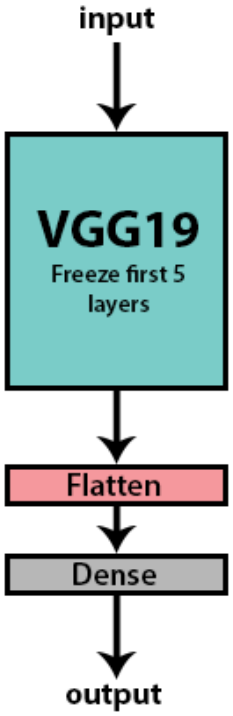

FIG 4 SOLUTION 1 ARCHITECTURE

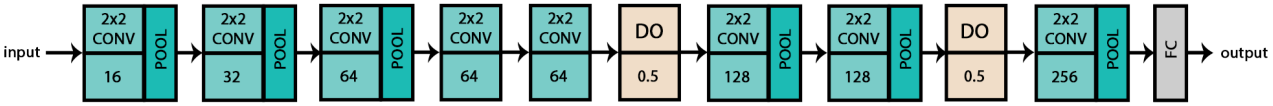

FIG 5 SOLUTION 2 ARCHITECTURE

Refinement

The initial model (first solution in figure 4) was trained on the non cropped images because after cropping the images I realized they were of different shapes and only got the idea to set the hight and width to be the same before cropping after training has almost finished on the full sized images. The first solution overfitted but it can possibly be improved by adding custom dropout layers and by decreasing the number of epoch (from 20 to 10) and increasing the batch size. Training on the cropped images will also speed the training time.

For the second solution (figure 5), the initial solution yielded a score that is not better than the benchmark. The number of epochs was 15 and the batch size was 20 and the kernel size was set to 3 x 3. It seemed that the accuracy and loss metrics were stabilizing after the second or third epoch so I changed the kernel size to 2 x 2 and noticed a slight improvement.

At the second trial I increased the batch size and the number of epochs and set both to 50. There was noticeable improvement but the model overfitted and did badly on the testing set. I added two dropout layers to the CNN to reduce overfitting and trained the model again.

One of the trials which yielded a result slightly better than the benchmark had batch size of 20 and number of epochs was set to 15.

After several trials I noticed that increasing the epochs to 50 or more and batch size (100, 150, 200) always caused the model to overfit (even with regularization) so for the final trial, I set the number of epochs to 20 and the batch size to 20 and this yielded results better than the benchmark but the score was still relatively low.

IV. Results

Model Evaluation and Validation

The final model was selected according to the evaluation metric, it yielded the lowest loss rate after many trials. The number of epochs is set to 20 and the batch size is set to 20.

When tested with the images in the dataset the model performed poorly in identifying specific whales, however it did well in predicting whales that are similar in shape and position as can be seen in figure 6. This indicates that in order to achieve the best accuracy, the images should be cropped to focus only on the head mark (to not confuse water splashes with head marks) and then the heads should all be aligned to one direction.

One of the things that I noticed is that the whale ID that has the largest number of images (whale_95370) got predicted on many test predictions which could indicate that it is an outlier in this dataset (has a total of 47 images in the dataset) and that this imbalance in the dataset is majorly affecting the performance of the model.

The model is not robust and cannot be trusted to make accurate predictions especially for the classes with very small number of images. This model needs further improvement in order to achieve the desired solution of fully automating the process of whales detection.

Justification

The final solution provides better results than the benchmark but it does not particularly solve the problem. The benchmark yielded a loss of 5.99 while the final model yielded 5.90. The final solution does a little better than random chance on predicting the whales but not well enough to depend on entirely in the classification process.

This solution will not automate the full process of detecting and cataloging whales but it can aid that process by providing recommendations. I think solving this problem requires a really deep neural network with good regularization. The model that was built in this solution was not deep enough to recognize the head marks, it was only able recognize the shape of the whale and in some cases recognize specific unique whales. When more layers were added to deepen the neural network the model tended to overfit so the best approach to resolve this problem is to preprocess the image and balance the dataset.

V. Conclusion

Free-Form Visualization

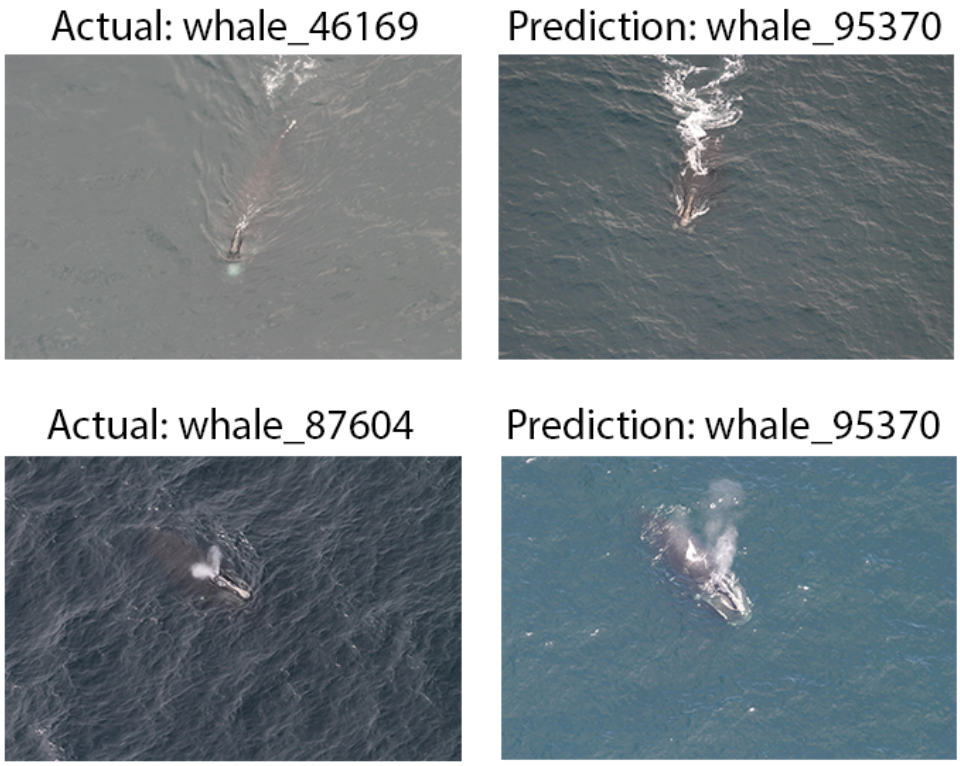

FIG 6 VISUALIZATION OF THE MODEL'S PREDICTIONS

Reflection

The full process can be summarized in the following steps:

1. Select the problem domain and find the dataset.
2. Preprocess the data.
3. Implement the model's architecture.
4. Train the model with the preprocessed data.
5. Evaluate the model and test its performance.

The most challenging and equally interesting part about implementing this solution was dealing with the dataset and trying to come up with the best way to preprocess it without losing the important features. The final model does not performed very well but I think the architecture will do better on a more distributed dataset and it can definitely be used to solve image classification problems.

Improvement

The model could definitely improve further either by attempting to balance out the dataset or by enhancing the model architecture. One of the winning teams created three CNNs, the first one detects the whale's head, the second one aligns the head and the last one classifies the whale[12]. I was unable to apply this solution because I did not know how to extract and label images of the head to train a network to recognize them and align them. Overall, preprocessing the images is the key to improving this solution.

References

[1] Kaggle, Right Whale Recognition, https://www.kaggle.com/c/noaa-right-whale-recognition
[2] Nur Shazwani Kamarudin, Mokhairi Makhtar, Syed Abdullah Fadzli, Mumtazimah Mohamad, Fatma Susilawati Mohamad, Mohd Fadzil Abdul Kadir, Comparison Of Image Classification Techniques Using Caltech 101 Dataset, 10 January, 2015, http://www.jatit.org/volumes/Vol71No1/9Vol71No1.pdf
[3] Yann LeCun, Yoshua Bengio, Geoffrey Hinton. Deep learning, 28 May 2015, https://www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf
[4] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
[5] James D. McCaffrey, The Difference Between Log Loss and Cross Entropy Error, March 31, 2017, https://jamesmccaffrey.wordpress.com/2017/03/31/the-difference-between-log-loss-and-cross-entropy-error/
[6] Wikipedia, Log Loss, http://wiki.fast.ai/index.php/Log_Loss
[7] Andrew B. Collier, Making Sense of Logarithmic Loss, December 14, 2015 http://www.exegetic.biz/blog/2015/12/making-sense-logarithmic-loss/
[8] How do convolutional neural networks work?, https://www.quora.com/How-do-convolutional-neural-networks-work
[9] Jason Brownlee, How To Improve Deep Learning Performance, September 21, 2016, https://machinelearningmastery.com/improve-deep-learning-performance/
[10] Eduardo Flores Github, https://github.com/eduardofv/whale_detector/
[11] Felix Lau, Recognizing and Localizing Endangered Right Whales with Extremely Deep Neural Networks, January 8, 2015 http://felixlaumon.github.io/2015/01/08/kaggle-right-whale.html
[12] Rober Bogucki, Which whale is it, anyway? Face recognition for right whales using deep learning, January 16, 2016 https://blog.deepsense.ai/deep-learning-right-whale-recognition-kaggle/