Happywhale - Whale and Dolphin Identification

Matthew Harding

April 2024

1 Introduction

This project explores the problem of classifiying images of dolphin and whales to identify indivduals. This work is based on the *Happywhale - Whale and Dolphin Identification* Kaggle competition [2].

Data for this competition contains images of over 15,000 unique individual marine mammals from 30 different species collected from 28 different research organizations. Individuals have been manually identified and given an individual_id by marine researchers.

Unlike a typical classification problem in which there is a fixed set of classes which examples of all classes within the training data, this problem required the ability to classify individuals not contained within the training dataset.

2 Training Data

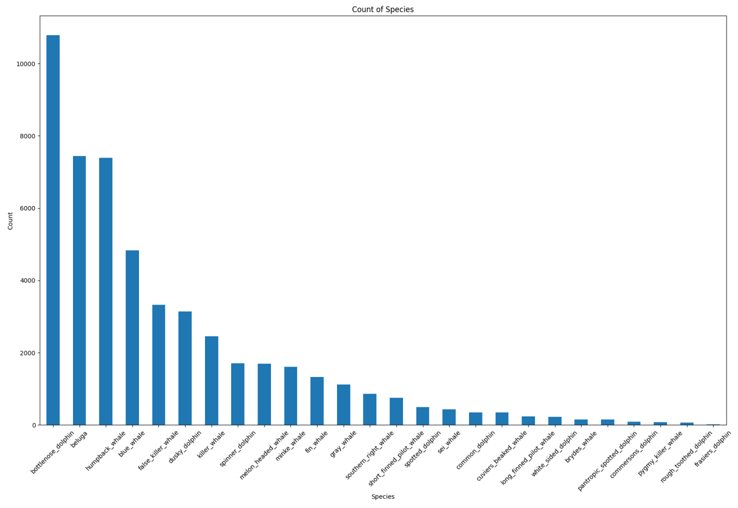

Figure 1: Number of training images per species

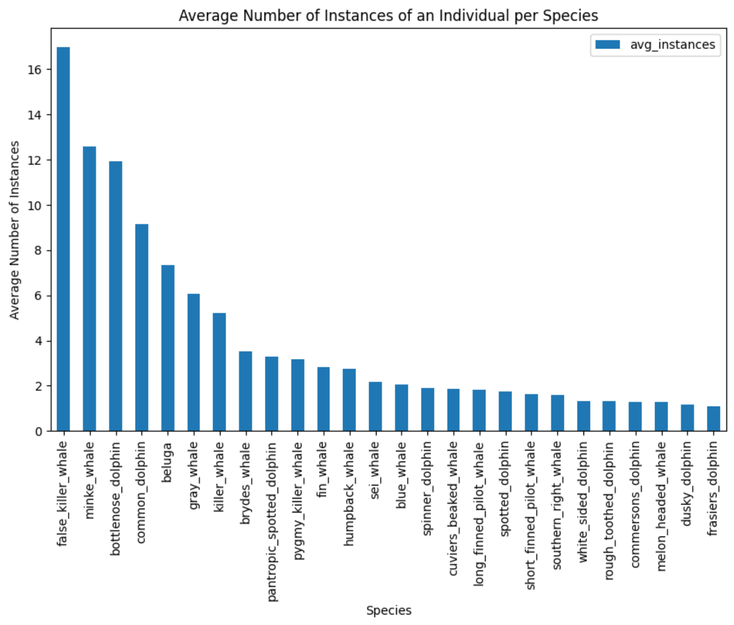

Figure 2: Mean count of images per individual by species

Looking at the histogram of species Figure 1, there is a clear inbalance in the number of images per species with the vast majority of labelled images being for bottlenose dolphins, beluga whales, humpback whales and blue whales. For some species the number of examples it so low it will be difficult to gain a high level of accuracy within classification.

Figure 2 shows the mean count of images per indivdual by species where we see another imbalance with some species on average containing a large number of images per indivudal whilst others only contain one or two images per individual.

3 Image Preprocessing

The training images came in a variety of sizes. Figure 3 shows an example unprocessed image. Large amounts of each image are just of water which is irrelvant to the classification task. We are only interested in the parts of the images that contain the animals.

A solution to this issue was proposed within the discussions of the Kaggle competition [1]. YoloV5 was trained on a dataset of 1200 pictures of whale flukes and the corresponding location of points on the edge of the fluke for those pictures [3]. This was then assumed to generalise to finding the bounding box for any part of a whale or dolphin within the water. However, it is worth noting that finding the bounding box of any part of the animal is considered an Out of Distribution problem. From this technique a cropped dataset was

Figure 3: Example image of a bottlenose dolphin

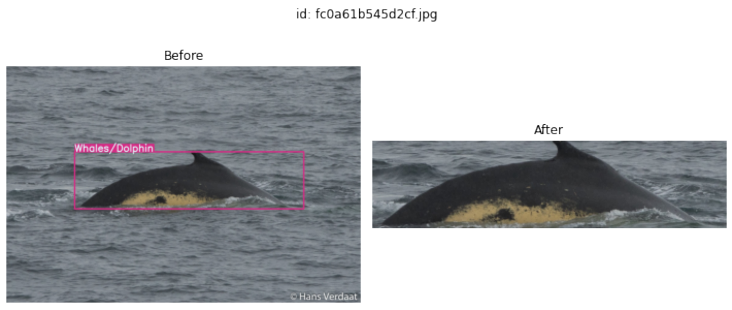

Figure 4: Animal detection using YoloV5 model

created with all images having a dimension of 256 x 256. This is the dataset that was then used to train the whale and dolphin classifier.

4 Image Transformation

A custom PyTorch dataset class called *WhaleDataset* was created for loading in the training images. One-hot encoding was used for the image lables, representing the individual ids provided.

The images were converted to RGB format as this was the compatible format for PyTorch. The images were then transformed using the TorchVision transforms package. The images were already all the same dimension from the preprocessing outlined in the previous section. The transformation involved convertsing the PIL image to a PyTorch tensor, scaling ther pixel values to the range [0,1] followed by normalizing the tensor by subtracting the mean and dividing by the standard deviation.

Scikit Learn's *train_test_split* was used to split of 80/20 was used. A random state was specified for reproducability between training runs. Shuffling was enabled on the test set.

5 Measuring Model Performance

Performance of the model was measured via the Mean Average Precision @ 5 (MAP@5). Instead of using just accuracy as a measure, MAP@5 was used to account of the complexity of the classification task. Accuracy only considers whether the top predicted class is correct or not. It does not take into account the ranking of predictions. MAP@5, on the other hand, considers the ranking of predictions within the top 5 predicted classes. It rewards models that rank the correct classes higher in the prediction list.

$$MAP@5 = \frac{1}{U} \sum_{u=1}^{U} \sum_{k=1}^{\min(n,5)} P(k) \times \mathrm{rel}(k)$$

- `U` is the number of images
- `P(k)` is the precision at cutoff `k`
- `n` is an indicator function equaling 1 if the item at rank `k` is the correct label, zero otherwise

6 Network Architecture

As this was an image classification task, a network architecture based around Convolutional layers seemed logical.

The first layer was a 2D Convultional layer. This layer takes the input image with 3 channels (RBG) and applies 32 different 3x3 convolutional filters to extract low-level features like edges, corners and simple patterns. The stride of 1 ensures that the filters are applied to every pixel, and the padding of 1 preserves the spatial dimen

Next, a ReLU activation function was run to add non-linearity to the model (????)

Finally a pooling layer was added.

MATT - Why do we then do two additional convolutional layers? Look at forward method, we reuse relu and maxpool

MATT - Is fully connected 2 considered a SOFTMAX?

7 Model Training

Despite the massive reduction in file size that came from using the cropped dataset instead of the original dataset, training times on the local machine was still prohibitively long. Therefore, training was done using AWS Sagemaker.

8 Conclusions

References

[1] Happywhale: Cropped data prepare yolov5. Website.

[2] Happywhale - whale and dolphin identification. Website.

[3] Whale fluke location labelled dataset. Website.