A Fluke in the Data: Humpback Whale Recognition

Marc Negre (marc7@stanford.edu) and David Zweig (dzweig@stanford.edu)

Stanford University CS230

The Problem

Commercial whaling, climate change, and industrial overfishing have devastated humpback whale populations. Marine biologists would like to track sightings of individual whales so that they can build an accurate picture of the movement of whale populations worldwide. We built a model to assist them by:

- Taking a picture of a whale fluke
- Passing the picture into a convolutional neural net
- Predicting which individual whale the tail fluke corresponds to

Data

The dataset comes from Kaggle's Humpback Whale Identification Challenge. The images in it are pooled from photographers around the world, and vary widely in their quality, color, and angle. They are labeled with the ground truth. Several factors made this an especially challenging dataset:

- There are 4,201 unique classes
- One class, 'new whales,' aggregates images for all unidentified whales and constitutes 8% of the training dataset
- 2,200 classes (> 50%) contain only one sample
- Images are a mix of 3-channel RGB and grayscale

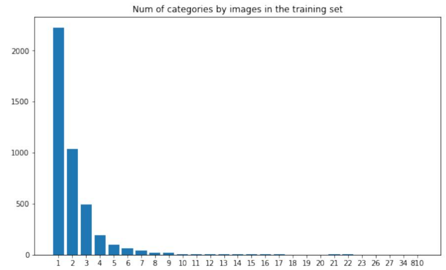
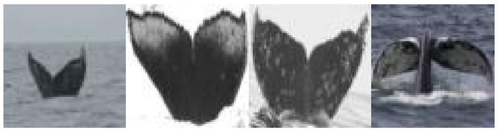


Figure 1: Samples per Class [1]



General Approach

Due to the sparsity of the dataset, we augmented our training dataset with a series of random rotations, shifts, and zooms. We then used this data to train on two different models and compared the augmented vs. non-augmented results.

Softmax Model

We implemented a 4,250-class softmax model using cross-entropy loss. Because the Kaggle challenge allowed us to output 5 predictions per sample, we hard-coded 'new whale' as an output and optimized for the other 4 predictions. The formula for cross-entropy loss is:

$$-\sum_{i=1}^{m} p(x^i)\ log\ q(x^i) \tag{1}$$

Triplet Loss

We also trained a softmax model utilizing triplet loss rather than cross-entropy loss. Triplet loss is expressed as follows, where a corresponds to an anchor image, p to a positive sample, and n to a negative sample [2]:

$$\sum_{i}^{N}[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha]_+$$

Results

Experiment \Accuracy	Train	Dev	Test
Softmax, alpha=0.0001	0.296	0.285	N/A
Softmax, alpha=0.001	0.334	0.285	0.3262
Softmax, alpha=0.01	0.322	0.268	N/A
Softmax, alpha=0.001, Augmented Dataset	0.393	0.306	0.3263
Triplet Loss attempt	0	0	N/A

Discussion

As two non-CS students with little coding experience, one of the biggest lessons from this project was a newfound understanding of the extensive backend work required to actually deploy machine learning models in the wild. We spent so much time building our workspaces and data pipelines that we were left with much less time to tweak our models than we would have liked. That said, we did come away with some solid insights from our models:

- Our error analysis found that 3 classes were present in >90% of our predictions. Our models ended up optimizing based on probability rather than distinguishing features.
- Our results were in the middle of the pack for participants in the Kaggle challenge
- Data augmentation cannot overcome extreme sparsity when training triplet loss

Lessons Learned and Future Paths

With a further six months to work on the project, we would focus our efforts on improving our data augmentation process and on implementing transfer learning. In particular, we are interested in:

- Implementing cross-validation, so that we can fully utilize our sparse training set when making predictions on the test set
- Deploying on-the-fly data augmentation methods, such as the Keras Image Preprocessing layer
- Using the pre-trained Facenet model and training one or two new fully-connected layers at the end to tune it for whale flukes instead of human faces
- Expanding our hyperparameter optimization

References

[1] Lex Toumbourou.
Humpback whale id: Data & aug exploration.
Available at `https://www.kaggle.com/lextoumbourou/humpback-whale-id-data-and-aug-exploration/notebook`(2018/03/21).

[2] Florian Schroff, Dmitry Kalenichenko, and James Philbin.
Facenet: A unified embedding for face recognition and clustering.
In *CVPR*, pages 815–823. IEEE Computer Society, 2015.