# CS583A: Course Project

Saeid Hosseinipoor

May 17, 2019

## 1 Summary

I participate an inactive with late submission competition of Humpback Whale Identification based on photos. The final model I choose is Open Face, a deep convolutional neural network architecture, which is a Siamese network of a variant of Inception model takes $192 \times 192$ images and outputs feature vector of the image. To predict the class of test image we find nearest neighbors of the feature vector in the same space of training set outputs. I implement the convolutional neural network using Keras and run the code on a PC with one Intel i7 CPU and 64 GB memory. Performance is evaluated on the top-5 classification accuracy metrics. In the public leaderboard, my score is 0.23182; I rank 2023 among the 2131 teams. In the private leaderboard, my score is 0.24015; I rank 2025 among the 2131 teams.

## 2 Problem Description

**Problem.** The problem is to identify humpback whales on photos. This is a binary classification and image classification problem in the same time. The competition is online at `https://www.kaggle.com/c/humpback-whale-identification`. This problem is very similar to face recognition problem explained in the course. We have photos of some whale's tails. Most of the photos are tagged by a given tail to a whale. Some of them are labeled as unknown. The assumption is the whale tails are unique like a person's face and we can recognize a whale by looking at its tail.

**Data.** The data are arbitrary JPEG images. Image sizes are not same neither their width, height, nor channels. Some of them are colorful, some are grayscale and sizes vary from 100 pixels to 2000 pixels or more. The number of training samples is about $n = 25,000$. The total number of classes including unknown is 5005. The training set is not well-balanced.

**Challenges.** There are lot of challenges here: First of all, We have imbalanced data set. the number of classes respect to instances are too much. We have 5005 different classes in 25000 instances. If even we consider it as balanced data set, we have average of 5 instances per classes. In reality, some classes just have one instance but some of them like unknowns have tens of them. Other challenge is that the amount of the classes; like face recognition problem, we need to classify photos as individual whales. This huge amount of classes make it difficult to use softmax as a classifier. Therefore I decided to use Siamese network [**?** ] to classify triplets. Another challenge is the variety of inputs. Images are not in same size or even in same aspect ratio. Some are colorful and some are gray scale.

## 3 Solution

**Model.** The model I finally chose is the Open Face Recognition[**?** ], a deep convolutional neural network architecture, which is a variant of Inception model [**?** ]. It takes $192 \times 192$ images and outputs a $128-dimensional$ feature vector. I used this network to compare and distinguish similar and non-similar pairs of images. To predict the class of test image we find nearest neighbors of the predicted feature vector in the space of training set. A description of Keras Open Face is online: `http://krasserm.github.io/2018/02/07/deep-face-recognition/` and the open source code is also available online: `https://github.com/iwantooxxoox/Keras-OpenFace`.

**Implementation.** I implement the Open Face model using Keras with TensorFlow as the back-end. My code is available at `https://github.com/saiedhp/CS583A-2019Spring/blob/master/project/humpback-whale-identification/kernel7631789da2.py`. I run the code on a PC with one Intel i7 CPU and 64 GB memory. It takes 20 hours to train the model. Model input is a triplet of images including anchor, positive and negative instances. The anchor is an image with a known class, positive is a different image in the same class as anchor, and negative is a different image in different class. I select the anchor image randomly from known classes (excluding unknown class), which have more than one member. Finally triplets of $(I_{anchor}, I_{positive}, I_{negative})$ passes to the model which a pair of $(I_{anchor}, I_{positive})$ fed into the network and another pair of $(I_{anchor}, I_{positive})$ fed into the same network with same weights. The similar pairs should be as much as possible close together while the non-similar pairs should be further in the feature space. After training, I hoped the network can discriminate between the images by producing a feature vector. The most 5 nearest vector to the target vector have more chance to be a correct answer. I also tried to find a threshold to judge about the unknown class or unseen whales. The assumption is if I can not find a sample close enough to the target vector, it is better to classify it as an unknown.

**Settings.** The goal of the training is to increase the $L_2$ distance between non-similar classes and decrease the distance of similar classes in same time. The definition of the loss function is:

$$L = \sum_{i=1}^{n} max[0, \|f(I_{anchor,i}) - f(I_{positive,i})\|_2^2 - \|f(I_{anchor,i}) - f(I_{negative,i})\|_2^2 + \alpha]$$

where $f(.)$ is the deep network which outputs the feature vector of input image, and $\alpha$ is the margin of this discrimination that I considered it as 1.0. The loss function is a combination of $L_2$ distance and hinge loss. The opitmizer is Adam. I train the model with batch of 200 for 50 epochs.

**Advanced tricks.** I used a pretrain model which was pretrained on human face. I expected it was a good point to start with high level feature extraction. It's true that human face is different to whale tails but it is also true that we need to extract basic feature to construct advanced features in middle and latest layers.

## 4 Compared Methods

Before this model, I tried to classify the images based on a deep convolutional network, but it failed. The number of classes made it impossible to classify the images by softmax classifier. My

final model is also awful but I don't have time to explore another methods and find better solution. Actually, this method is supposed to be good and I expected at least 80% accuracy. It is possible that the inaccuracy comes from some bug in code development phase. I doubt if the implementation is wrong or model design.

## 5 Outcome

I participated in an inactive competition with late submission. In the public leaderboard, my score is 0.23182; we rank 2023 among the 2131 teams. In the private leaderboard, my score is 0.24015; we rank 2025 among the 2131 teams.