The Nature Conservancy - Fisheries Monitoring

Team
Jayaram Akkinepally `jakkin2@uic.edu`
Ramavaishnavi Pemmaraju `rpemma2@uic.edu`
Kaushik Kompella `kkompe2@uic.edu`
Gayatri Gamya Patirla `gpatir2@uic.edu`

1 Problem description

Fossil records show that the arrival of the Homo sapiens has accelerated the process of extinction of terrestrial beings by occupying their habitat which did not stop with land. Latest reports proved that it is highly plausible to face a major extinction which would cause unprecedented damage to marine ecosystems.

Approximately 50 percent of the world's population depends on seafood for their main source of protein. In the Western and Central Pacific region, where 60 percent of the worlds tuna is caught, illegal and unregulated fishing practices are threatening marine ecosystems and global seafood supplies. The Nature Conservancy is working hard with regional and global partners to preserve this fishery for the future.

Currently, the conservancy is looking to the future by using cameras to dramatically scale the monitoring of fishing activities in order to fill compliance monitoring data gaps. Although these electronic monitoring systems work well, the amount of raw data produced is complex, incommodious and expensive to process manually. However, machine learning has the ability to solve the current problem. In order to facilitate this, the conservancy has started using the boat cameras to capture the images of the fish catch in various angles, which can thereby be used to detect the fish specie.

The main objective is to develop algorithms to detect and classify species of tunas, sharks, etc which will help to accelerate the video review process. Faster review and more reliable data will enable in reallocation of human capital to management and enforcement activities. We want to develop a solution to this image classification problem by developing a model which would predict top 5 maximum probability scores and thereby estimate the likelihood of fish species in each picture.

2 Data description

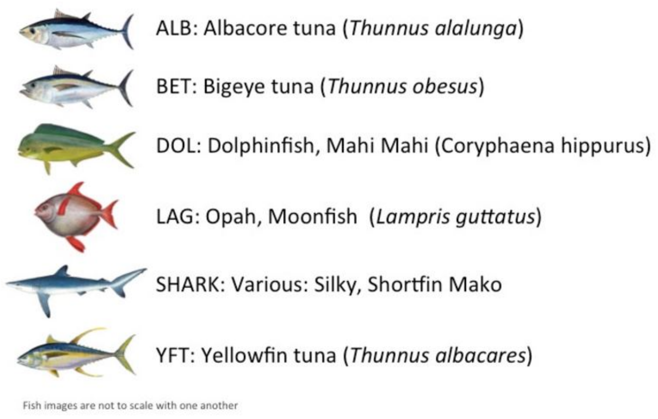
In this problem the data corresponds to the images of different fishes. Each image has only one fish category, except that sometimes there is a very small fish in the pictures that are used as bait. Images are captured from boat cameras of a fishing boat at different angles.

There are eight target categories available in this dataset: Albacore tuna, Bigeye tuna, Yellowfin tuna, Mahi Mahi, Opah, Sharks, Other (meaning that there are fish present but not in the above categories), and No Fish (meaning that no fish is in the picture).

The whole data is divided into the training set which comprises of 3777 images across 8 different categories and the test dataset which contains 1000 images. Each image is a color image of different size with varying pixels.

Note: The data set is a relatively small dataset with fewer samples in each category, considering the problem at hand.

The categories in pictorial form is as follows:


ALB: Albacore tuna (*Thunnus alalunga*)
BET: Bigeye tuna (*Thunnus obesus*)
DOL: Dolphinfish, Mahi Mahi (Coryphaena hippurus)
LAG: Opah, Moonfish (*Lampris guttatus*)
SHARK: Various: Silky, Shortfin Mako
YFT: Yellowfin tuna (*Thunnus albacares*)
Fish images are not to scale with one another

Challenges with the data

1) The images were captured from cameras which lack clarity.

2) There are 465 images without containing a fish. Considering the training dataset to be small, this is a huge number.

3) Even in the images where there are fishes, most of the time they are cornered, occluded or has problems with the location of the camera.

3 Proposed algorithms, tools and techniques

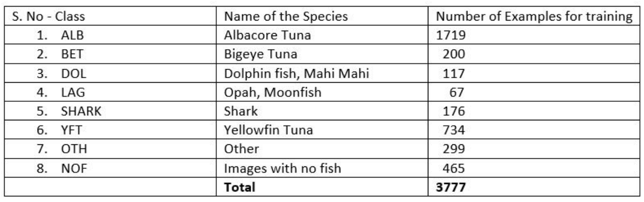
Tools Being Used: Python, Keras, Tensorflow etc. Concepts: CNN, VGG based clustering etc.

Best model can be evaluated with the multi-class logarithmic loss. Each image has been labeled with one true class. For each image, we must calculate a set of predicted probabilities (one for every image). The formula is then,

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

where N is the number of images in the test set, M is the number of image class labels, log is the natural logarithm, yij is 1 if observation i belongs to class j and 0 otherwise, and pij is the predicted probability that observation i belongs to class j.

The submitted probabilities for a given image are not required to sum to one because they are rescaled prior to being scored (each row is divided by the row sum). In order to avoid the extremes of the log function, predicted probabilities are replaced with max(min(p,1 - 10 power -15),10 power -15).



S. No - Class	Name of the Species	Number of Examples for training
1. ALB	Albacore Tuna	1719
2. BET	Bigeye Tuna	200
3. DOL	Dolphin fish, Mahi Mahi	117
4. LAG	Opah, Moonfish	67
5. SHARK	Shark	176
6. YFT	Yellowfin Tuna	734
7. OTH	Other	299
8. NOF	Images with no fish	465
	Total	**3777**

Pre-Trained models which can be used:

a) VGG-16

b) VGG-19

c) ResNet50

d) Inception V3

e) CRNN

Approach

With the current size of the training data, the problem at hand is quite challenging. The following approaches would be considered for the classification problem:

1) Training a Neural Network from the scratch

2) Using the features of a pre-trained network as an input to our hand built network

3) Changing the last layers of a pre-trained network (fine-tuning)

NOTE

We have built a very basic CNN from the scratch which gave us a loss value of 1.3 (Kaggle score board and a rank of 1301, team name: VVJ) with the predicted probabilities obtained and submitted. Although training a CNN on this small data set would yield reasonable results, considering the time available for this project, we would focus on implementing options 2 and 3 stated above.

Data pre-processing

1) Random transformations of the training examples to avoid overfitting (ensure the model doesnt see the same picture twice).

2) Locating/creating a boundary for our object of interest in the image.

3) Over/Under sampling in order to avoid unbiased results.

In this section of the project,

1) Initially, the reference papers for the above pre-trained models would be studied.

2) As all the above pre-trained networks are trained on ImageNet data, we will directly classify our data set on the above models and compare results.

3) Feature extraction would be performed for each image in the training dataset we will run the models on training and validation data and record output until before the fully connected layers into numpy arrays.

4) Features obtained from these pre-trained models would be used on simple fully connected neural networks with various activation functions such as: a. Linear

b. ReLU

c. Sigmoid

d. Tanh

e. Leaky ReLU

f. Maxout

5. Kaggle discussions and kernels would be studied and discussed with team on a weekly basis and best/relevant ideas would be considered for implementation.

6. If time permits, a classification model would be built from the scratch, details of progress and scores would be presented.

4 References and papers:

The study and research of following links help us to understand the importance of problem statement in a much better way.

Resources to understand and get familiarized with the problem statement:

http://www.conserveca.org/
http://www.conserveca.org/campaign/transforming-the-last-tuna-stronghold?c=1
https://kaggle2.blob.core.windows.net/competitions/kaggle/5568/media/TNC.mp4
https://www.kaggle.com/c/the-nature-conservancy-fisheries-monitoring

Papers:

1) VGG Model

https://arxiv.org/abs/1409.1556

2) ResNet Model

https://arxiv.org/abs/1512.03385

3) Inception V3 Model

https://arxiv.org/abs/1512.00567

References:

1) Simonyan, Karen, and Andrew Zisserman. ”Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:1409.1556 (2014).

2) He, Kaiming, et al. ”Deep residual learning for image recognition.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

3) Szegedy, Christian, et al. ”Rethinking the inception architecture for computer vision.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

4) https://github.com/fchollet/deep-learning-models

5 Possible Experiments and metrics of success

Accuracy, Class Recall, Specificity and AUC (ROC) of predictions of the category of fish are a few metrics of success we shall chase.

Planned Experiments include Models and Methods for

1) Improving the class recall for a specific kind of (more valuable or rare) fish.

2) Improving the prediction rate for fishes with less number of images in the training

data (unbalanced classes).

3) Improving the overall prediction accuracy.

6 Potential conclusions that could be derived from the project

Dependence of Prediction metrics for the 8 classes of fish on the following would have been studied and summarized:

1) Models used

2) Fine tuning methods over hyper parameters used

3) Methods for feature extraction used

4) Methods for increasing/improving training data set used

Apart from overall prediction, we would be able come out with the methods that gave best results while foucsing on a particular kind of fish.

7 Results obtained so far

Data Preprocessing:

1. Each image has been rescaled to 224 x 224 x 3 sized tensor.

2. All elements in the 3D matrix have been normalized (divide by 255)

Models Tried:

Random Forest

1. Each image has been vectorized. (vector shape is 224*224*3)

2. Random forest has been built on the resulting matrix.

3. The results prove that the model is an overfit.

Metrics

a. Accuracy on validation data: 94.06

b. Precision on validation data: 94

c. Recall on validation data: 94

Feed Forward Neural Network with Sigmoid activation

1. Each image is vectorized.

2. A 6-layer neural network with sigmoid activation is built

3. The model has been trained and validated across 20 epochs

Metrics

a. cross-entropy loss (best across all epochs) for train data: 1.3102

b. accuracy (best across all epochs) for train data: 58.58

c. cross-entropy loss (best across all epochs) for validation data:1.2659

d. accuracy (best across all epochs) for validation data: 58.22

Convolutional Neural Networks

ReLU activation

Cross Entopy Loss

Metrics [12pt] a. Cross entropy loss (best across all epochs) for train data: 0.7875

b. Cross entropy loss (best across all epochs) for validation data: 1.1870