Multi-class Image Classifaiton with Fisheries Monitering Data

Kewen Zhang, Pengfei Wang, Yueying Teng
Department of Statistics
Columbia University
Kz2246@columbia.edu Pw2406@columbia.edu Yt2495@columbia.edu

***Abstract*— This project is an ongoing Kaggle competition, we managed to achieve the top 10% on the public leaderboard at the end of the project. This report demonstrates everything we have implemented in the past weeks. There are around 5000 images in the dataset, including 4000 training images of 8 different classes and 1000 test images. Firstly, a Cascade Classifier was created to detect the location of fish and crop the fish from the training image. Secondly, we trained two different algorithms in fish classification. The first is Gradient Boosting Decision Tree, which was trained using Bag of Image Features. However, the result is not satisfying. Convolutional Neural Network model, consisting of four convolutional layers, some followed by max-pooling layers, and three fully-connected layers with eight-way softmax, was then used to produce a better prediction probability result. The best model used to produce submission result is a variant of this Convolutional Neural Network model.**

***Keywords*— Haar Cascade; Bag of Words; Gradient Boosting Decision Tree; Convolutianl Neural Network; OpenCV; Spark; Keras; Python**

I. INTRODUCTION

According to the studies, nearly half of the world depends on seafood as main source of protein. In the past, fish was more of a sustainable resource due to limitation in technology. However, in recent years, with the advent of technology in the fishing industry, overfishing has significantly affected many fisheries around the world.

Human consume fish at a much faster rate than the rate fish reproduce themselves. If this rate of consumption continues, certain species, for example, ALB (albacore tuna), LAG (Opah), Sharks (Various: Silky, Shortfin Mako), will become endangered or even extinct.

The depleted stocks and poor fisheries management cause great economic that is worth of billions of dollars. Apart from this, the balance in the ecosystem is broke due to the distinction of fishes in the food chain that cannot support the other fishes rely on them. These situations have been worsened in the recent decade, which calls for great attention in good fisheries management.

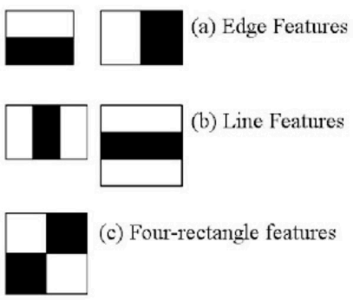
The Nature Conservancy is now working with regional and global partners to preserve fish species for the future. Cameras are installed on fishing ships to monitor the scale of the fishing activities. More specifically, with the image of fish caught captured by the cameras sending back, the support system will be able to distinguish the type of fish caught. If it is one of the endangered fish, signals will be send back to the fishing ship and proper action will be taken on time to protect these species.

The goal of this project is to build the support system that can distinguish 7 classes of fishes. Given the image sent back from the ship, the fish will be cropped out using image detection method, namely Cascade Classifier. Consequently, muti-class image classifier, Gradient Boosting Decision Tree classifier and Convolutional Neural Network model, can be trained using the fish images and the prediction for each image is the class of fish with the highest prediction probability.

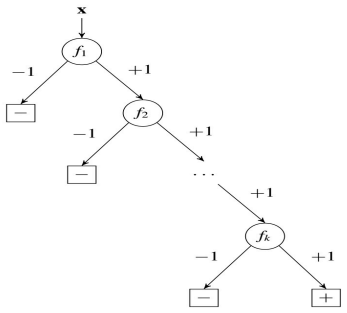
II. RELATED WORKS

A. Cascade Classifier

Haar Feature-based Cascade Classifiers is one of the most effective ways to detect objects in images, one widely used example is the face detection function implemented in cameras and smartphones. Haar Feature-based Cascade Classification method was first proposed by Paul Viola and Michael Jones in 2001 in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features".



Above is an example of Haar features drawn from the original images, each feature is a single value obtained by subtracting sum of pixels under white rectangle from the sum of pixels under black rectangle. At the beginning, for each image in the training set, a window is given for the algorithm to detect all types of features. For example, if a 24x24 window is give in the original image, more than 160,000 features can be generated in this small area. This is a time-consuming process, and it is always the case that the target object only accounts for a small area of the entire image, which means a large proportion of the calculations are meaningless. In order to decrease the computational burden, Cascade Classifiers is introduced here by setting a parameter named stage.

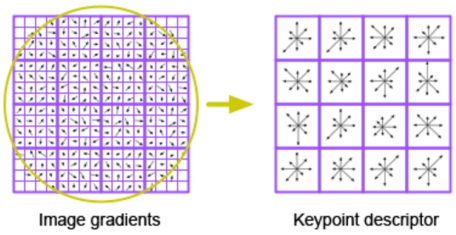


Above is an illustration of the Cascade Classifier. After all the Haar features are extracted from the original images, the algorithm moves through the entire area of the image using a window with predetermined size. If the area in the window is classified as -1 with Haar features, this area is classified as with no target object showing. When the classification result is +1, the area in the window considered as having the object presented. After going though all the pixels in the image. Eventually, the pixel areas that passes all the stages give the location of the target object.

B. Bag of Words

Bag of Words model was used to extract image features and quantify each training image as vector, which can be used further in the classifier training process as input.

First, keypoints on the object was extracted to provide a "feature description". There are several methods widely used in the industry to detect interest points: Edge Detection, Corner Detection, Blob Detection and Ridge Detection. The detection method is chosen based on the quality of the target objects in the picture. To perform accurate extraction of keypoints on testing images, features extracted from the training images should be detectable even under changes in image scale, noise and illumination. Therefore, SIFT features were suggested to be used here in detecting and describing interest points on all cropped training images.



Once features have been detected, a 4 x 4 pixel image patch around the feature can be extracted. For each image patch, a descriptor vector of dimension 128 was calculated for each pixel based on the 8 bins of image gradients orientations. The result is a matrix of features descriptors for all the images in the training set. Up till now, it is made sure that all the keypoints are invariant to image location, illumination, scale and rotation. The next step is to create an image vocabulary using a clustering method, for example, K-Mean Clustering. This is used to quantify training images as vectors, which can then be used in training Machine Learning Algorithms.

To quantify the training images, each feature from the image was individually compared to the visual dictionary and the candidate matching features were found based on Euclidean distance of their feature vectors. Accordingly, each image was represented by a histogram with K bins (the number of clusters K used in the K-Mean clustering) with each indicating the frequency of the appearance of the clustered features in the visual vocabulary.

In this way, each image is quantified by the features collected from all training images and further training of classifiers will be based on these vectors.

C. CNN

Convolutional neural network (CNN) is well known for many applications such as hand-written digit recognition, face recognition, image classification. It is a type of feed-forward neural network, which passes an input through one or more layers of neurons or nodes. These neurons each represents a linear combination of its input and a specific activation function will be used to pass these combinations. The outcomes are passed to the next layer and finally culminate d in an output layer. This algorithm can learn highly non-linear function, depending on the network structure. There are three

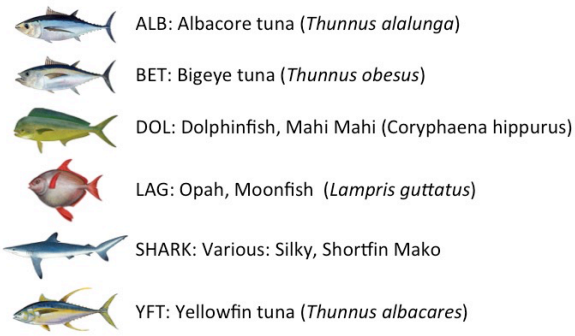
kinds of layers, convolutional layer, pooling layer and fully-connected layer.

Convolutional layers are the core structure of a CNN, which preserve the spatial relationship between pixels by using small squares filters on input data. Pooling layers are commonly inserted between successive convolutional layers, which combine the neuron clusters output. Fully-connected layers connect to all activations in the previous layer and produce a distribution over target class labels.

III. System Overview

A. Dataset Used

The dataset is downloaded from Kaggle website. There are around 5000 images totally, 4000 training images and 1000 test images. The training images are classified into eight categories, six of them are specific types of fish, one is Other Fish category and the another is No Fish category.



The project goal is to detect whether there are fishes appears in test image, and which species of fish they are if the image contains fish.

B. System Design

Our system consists of three parts as previously introduced. (1) **Haar Cascade**: The original image contains huge amount of noise in the background, which not related to the fish. In order to improve the prediction accuracy, fish detection cascade is trained to crop the fish images from original data. These cropped fish images are used in the following two parts, together with the original data. (2) **Bag of Words Model**: Because of the great performance in the field of image annotation and retrieval, Bag of Words is adopted to generate fish image features. Gradient boosting decision tree is implemented after the feature generation process. (3) **Convolutional Neural Network**: 2D CNN is chosen to improve the prediction result. The model consists of four convolutional layers followed by some max-pooling layers, and three fully-connected layers with eight-way softmax. CNN for some specific fish categories are built to re-weight the prediction outcome of the main model.

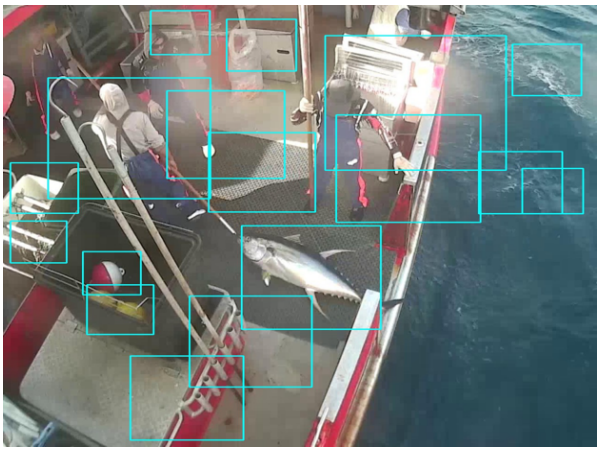
IV. Algorithm

A. Cascade Classifier

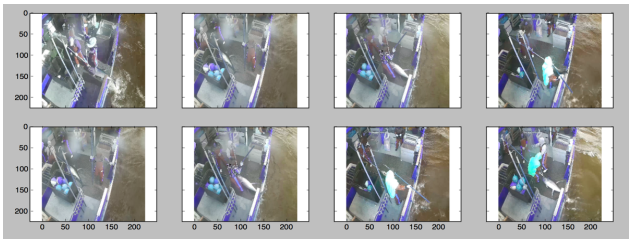
Instead of utilizing the raw images to build classifiers directly, fish were cropped from the original images first. This helps to eliminate noise in the background, including the waves, fisheries boats, crew member, which account for a large area of the image. Cascade Classifier is introduced in this part to detect the fish, which acts like the data cleaning process.

In the training process of Cascade Classifier, two groups of instances are considered, namely positive instances and negative instances. Positive instances are the cropped fish images, while negative instances are images without fish. In order to obtain clean positive instances, the fish location data for training images (provided by Nathaniel Shimoni) is used to crop fish from the background, while the negative instances are the same images but have fish replaced with other objects. Moreover, more negative instances were added to this training data set for Cascade Classifier. These negative instances are images of shipping crews and boxes on the ships, which appear frequently in the background that can cause confusion for the classifier. The first Cascade Classifier for fish detection was built using the original images.

The result from this first Cascade Classifier is not satisfying. More than one objects that are not fish are returned by the algorithm. The following is an image to show the result from this first Cascade Classifier.



To increase the accuracy in fish detection, a second Cascade Classifier was built to classify fishing ships. As the location of cameras installed on different types of ships are fixed, the pictures taken from theses cameras capture fish at a relatively fixed location on the image due to the fixed camera angle. If the type of fishing ship is known, the relatively fixed location of the fish is known, and the second Cascade Classifier can be trained focusing on these smaller areas to detect an exact location of fish.



Above is the result of one type of fishing ship from the clustering. The clustering was made possible by using DBSCAN, density-based spatial clustering of applications with noise. According to this clustering result, there are 27 types of fishing ships and 3 outliers in the training data. Furthermore, the location of the smaller possible area for fish is decided for each type of ship and these locations are saved in json file for further steps.

The second Cascade Classifier was built by using the same process as the first one. The only difference is that this classifier was trained using the smaller area of the original images saved in the json file instead of the original images. The following image shows the detection result using the second Cascade classifier. It can be see that the exact location of the fish is detected by the blue window with the minimum size.

B. Gradient Boosting Decision Tree

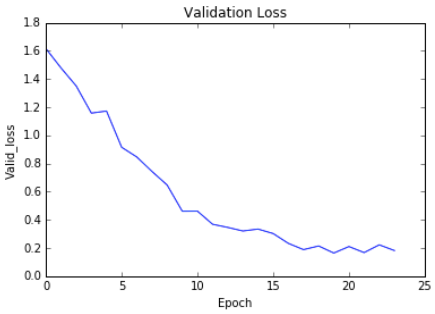
By using SIFT feature extractor and descriptors, it is made sure that all the keypoints are invariant to image location, illumination, scale and rotation. After all training images were processed using the same SIFT descriptors in producing the image vocabulary, a 4465 x128 feature descriptor matrix is achieved and an image dictionary of 500 clusters was then created based on this 4465 x128 feature descriptor matrix using K-mean Clustering.

By using the 500 clusters image vocabulary created by Bag of Words model, each training image can be quantified by clustered feature vectors. Each feature in the training image is matched individually back to the clusters in the image vocabulary using L2_Norm in OpenCV together with Python. This gave a histogram feature vector of 500 bins for each training image.

These histogram feature were used to train a Gradient Boosting Decision Tree classifier in Spark. Two parameters, number of trees and tree depth were tuned using 5-fold Cross Validation to achieve the best classification results.

C. Convolutional Neural Network

As the above graph shows, the CNN contains nine layers; four of them are convolutional layers, two of them are pooling layers and remaining three of them are fully-connected layers. The first convolutional layer filters the 64x64x3 input image with 16 kernels of size 3x3 and the second convolutional layer has 16 kernels of size 3x3. The third convolutional layer takes the pooled output of the second convolutional layer and filters it with 8 kernels of size 3x3. The fourth convolutional layer has another 8 kernels of size 3x3.The first two fully-connected layers output the array of shape (*, 96) and (*, 16). The output of final fully connected layer is fed to 8 classes softmax. The ReLU (Rectified Linear units) is adopted in all the convolutional and filly connected layer structure.



Log-loss in 30 epochs for one validation fold

Log-loss is utilized to evaluate the model prediction performance. After parameter tuning, training algorithm runs 30 epochs in each chosen 20-folds cross validation. The training process will be terminated when no improvement shown in log-loss for 3 consecutive epochs. The validation and prediction output are the average of these 20 models outcome.

V. SOFTWARE PACKAGE DESCRIPTION

A. *Cascade Training using OpenCV*

1) Set necessary enviroment variable to ensure OpenCV works properly.
2) Build the working directory, input the prepared positive and negative images.
3) Collect images file name and create samples from positive images
4) Use OpenCV to train the fish cascade classifier and ouput the .xml file.

```
data — opencv_traincascade -data data -vec fish.vec -bg neg.info -numStages 10 -numPos 450 -numNeg 465 -w 128...
(opencv) dyn-129-236-224-163:data pfwang$ opencv_traincascade -data data -vec fish.vec -bg neg.info -numStages 10
 -numPos 450 -numNeg 465 -w 128 -h 128 -featureType LBP
PARAMETERS:
cascadeDirName: data
vecFileName: fish.vec
bgFileName: neg.info
numPos: 450
numNeg: 465
numStages: 10
precalcValBufSize[Mb] : 1024
precalcIdxBufSize[Mb] : 1024
acceptanceRatioBreakValue : -1
stageType: BOOST
featureType: LBP
sampleWidth: 128
sampleHeight: 128
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100

===== TRAINING 0-stage =====
<BEGIN
POS count : consumed   450 : 450
NEG count : acceptanceRatio    465 : 1
Precalculation time: 28
+----+----------+---------+
|  N |    HR    |    FA   |
```

B. *GBDT using PySpark*

1) Set necessary environment variables for Spark and Jupyter Notebook.
2) Configure Spark running on Jupyter Notebook (This step is somehow very complicated and time-consuming).
3) Use spark built-in Mllib to build the model.

C. *CNN using Keras*

1) Set necessary environment variables for Python2.7 (Warning: don't use Version 3.5)
2) Install relevant dependencies like numpy, scikit-learn, pillow, etc.
3) Install updated version of Theano and Keras.

```
Start KFold number 1 from 20
('Split train: ', 3591, 3591)
('Split valid: ', 189, 189)
Train on 3591 samples, validate on 189 samples
Epoch 1/30
34s - loss: 1.8075 - val_loss: 1.6243
Epoch 2/30
31s - loss: 1.6321 - val_loss: 1.5212
Epoch 3/30
30s - loss: 1.5657 - val_loss: 1.4387
Epoch 4/30
30s - loss: 1.5054 - val_loss: 1.3490
Epoch 5/30
30s - loss: 1.4292 - val_loss: 1.1980
Epoch 6/30
30s - loss: 1.3123 - val_loss: 1.0601
Epoch 7/30
31s - loss: 1.2158 - val_loss: 0.9686
```

VI. EXPERIMENT RESULTS

A. *Gradient Boosting Decision Tree*

The best model gave a 5-fold Cross Validation accuracy estimate of 74% after tuning two parameters, number of trees and depth of each tree. The same process of extracting features and transforming them to descriptors was applied on all the cropped fish in testing images. Then the descriptors were matched back to the 500 visual dictionary to quantify each testing image into feature vectors. After deriving the histogram feature vectors, all the histogram feature data was used in the classification with GBDT model and the prediction results were submitted to Kaggle, which was not satisfying enough. As a result, a more advanced model, Convolutional Neural Network model was proposed to be used in the further classification work.

```
5-fold validation resutls:
[ 0.75111607  0.76510067  0.76035834  0.76318743  0.667789  ]
Cross validation result:
0.741510303242
```

B. *Convolutional Neural Network*

The validation result after parameter tuning shows satisfactory results in some image categories, like ALB (Albacore tuna), BET (Bigeye tuna) and NoF (No fish).

image	ALB	BET	DOL	LAG	NoF	OTHER	SHARK	YFT
img_00005.jpg	0.0225	0.0027	0.0018	0.0013	0.9558	0.0042	0.0025	0.0092

NoF Category Prediction Probability 0.9558

However, for some other classes, like LAG (Moonfish) and DOL (Dolphinfish), the model performances relatively inaccurate prediction on those categories. One reason. In order to improve the mode accuracy, the cropped fish image obtained before can be used to classify specific fish categories. The new prediction outcome is used to reweight the prediction of the main model.

For example, the image shown above is a Moonfish, of which the original prediction probability is 0.2525.

image	ALB	BET	DOL	LAG	NoF	OTHER	SHARK	YFT
img_00071.jpg	0.3790	0.0643	0.0180	0.2525	0.0431	0.1706	0.0152	0.0572

Original prediction 0.2525

After training specific convolutional neural network for this fish category, the original prediction output is reweighted by the new model outcome.

image	ALB	BET	DOL	LAG	NoF	OTHER	SHARK	YFT
img_00071.jpg	0.0541	0.0092	0.0026	0.8932	0.0062	0.0244	0.0022	0.0082

Reweighted prediction 0.8932

We managed to achieve top 10% (54 out of 710) on the Kaggle leaderboard.

VII. Conclusion

The final classification results using the combined probabilities from the two Convolutional Neural Network models are satisfying for all 8 classes of images according to the submission ranking from Kaggle. Massive amount of time was invested in the trial and error process in this project. It started from building a Gradient Boosting Decision Tree Classifier using Bag of Image Features; and with gradual exploration, a Convolutional Neural Network model was proposed. In order to improve certain poorly classified class of fish, a second Convolutional Neural Network model was built to classify moonfish from the rest, and in combination with the original CNN model the prediction probabilities on the testing image data achieved top 10% ranking on Kaggle.

Everyone contributed equally to this project, which is divided into three parts generally: training Cascade Classifier with Haar Features, building Bag of Words model and Gradient Boosting Decision Tree classifier and training the two Convolutional Neural Network models.

It is believed that the project can be improved in terms of both the algorithm and the size of the dataset. For the certain types of fish that are poorly classified, the third or even the fourth Convolutional Neural Network model can be built to reweight the final prediction probabilities. As for the training dataset, more positive images could be scraped from the Internet. This helps to proved more detailed information of the seven classed of fishes without too much background noise. With the enlarged training dataset of high quality images, the prediction results can be further improved.

Acknowledgment

The cropped fish location in training data is provided by Nathaniel Shimoni, thanks for the great work!

References

[1] Myers, R.A. and Worm, B., 2003. Rapid worldwide depletion of predatory fish communities. Nature, 423(6937), pp.280-283.

[2] Bradski, G. and Kaehler, A., 2008. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.".

[3] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

[4] Read, J., Pfahringer, B., Holmes, G. and Frank, E., 2011. Classifier chains for multi-label classification. *Machine learning*, *85*(3), pp.333-359.

[5] Krizhevsky, A. and Hinton, G., 2009. Learning multiple layers of features from tiny images.

[6] Csurka, G., Dance, C., Fan, L., Willamowski, J. and Bray, C., 2004, May. Visual categorization with bags of keypoints. In Workshop on statistical learning in computer vision, ECCV (Vol. 1, No. 1-22, pp. 1-2).

[7] Read, J., Pfahringer, B., Holmes, G. and Frank, E., 2009, September. Classifier chains for multi-label classification. In Joint European

Conference on Machine Learning and Knowledge Discovery in Databases (pp. 254-269). Springer Berlin Heidelberg.

[8] Lienhart, R. and Maydt, J., 2002. An extended set of haar-like features for rapid object detection. In Image Processing. 2002. Proceedings. 2002 International Conference on (Vol. 1, pp. I-900). IEEE.

[9] Simard, P.Y., Steinkraus, D. and Platt, J.C., 2003, August. Best practices for convolutional neural networks applied to visual document analysis. In ICDAR (Vol. 3, pp. 958-962).