The Nature Conservancy Fisheries Monitoring

--Fish Type Identification

1. Introduction

Hundreds of millions of people worldwide rely on seafood as their main food source. In 2014, the world consumed over 146 billion kilograms' fish. Unfortunately, over 30% of the commercial fish stock were evaluated unsustainable fished ones. This percentage tripled compared to the statistics in 1974 [*FAO*, 2016]. The Western and Central Pacific provides 60% of the world's tuna stocks[*Aqorau*, 2001]. However illegal, unreported and unregulated fishing activities are causing damage to the sustainability of marine ecosystems, and putting global fish supplies in peril [*NIC*, 2016]. For mitigating the deteriorating problem, it is necessary to monitor the fishing practices globally. Using camera to monitor the fishing activities is an ongoing effort, which currently requires a tremendous amount of labor to identify fish types. Therefore, the machine learning comes to play to automate the fisheries monitoring process. Machine learning algorithms allow a model to learn to identify various kinds of fish captured by the fishing businesses; thus to monitor the fishing activity consistently. As an environmental engineer, it is particularly interested for me to harness the power of the new technology for solving a real-world problem. The interdisciplinary effort is expected to help nature conservationists to build a sustainable world for human beings.

The problem to be solved in this report is a machine learning competition project held by Kaggle. The goal of this project is to build a model to identify six fish types, which include Albacore tuna (ALB), Bigeye tuna (BET), Dophinfish, Mahi Mahi (DOL), Opah, Moonfish (LAG), Shark (SHARK), and Yellowfin tuna (YFT). In addition, there are two more categories: no fish (NoF), which means there is no fish in the images; and Other (OTHER), which means that there are fish in the images but not included in the six categories above. In total, there are eight classes for training and predicting: 'ALB', 'BET', 'DOL', 'LAG', 'Shark', 'YFT', 'NoF' and 'OTHER'. This is a typical image classification problem. A model employing convolutional neural network has the potential to tackle it. The model performance can be evaluated by the F1 score on a test dataset. A well-trained model can be used on fishery monitoring worldwide, to track fishing activities involving the aforesaid six types of fish.

In the following sections, dataset summary is described section 2. Section 3 articulated methods that are employed in this project. Section 4 reveals the results of the project. Finally, section 5 sums up the conclusions.

2. Dataset

2.1 Dataset characteristics

The dataset, provided by The Nature Conservancy, Satlink, Archipelago Marine Research, the Pacific Community, the Solomon Islands Ministry of Fisheries and Marine Resources, Australia Fisheries Management Authority and the governments of New Caledonia and Palau all contribute is available for the public on the webpage under Kaggle competition “The Nature Conservancy Fisheries Monitoring”. Among the 5 files listed on the page, only ‘train.zip’ includes both images and corresponding labels. Therefore, it is the dataset to be used in this project, for model training, validation and testing. In the dataset, there are total 3,777 images for the 8 classes. The number of images for each class is extremely imbalanced, see Fig. 1. Among the training data, ALB accounts for approximately 45%, and YFT accounts for approximately 20%. All other six classes account for the rest of 35%.

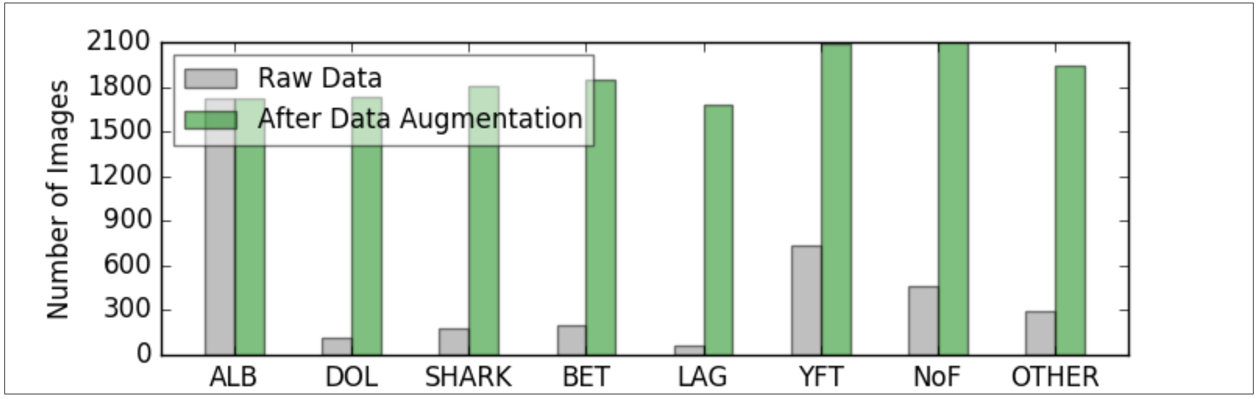


Fig. 1 Number of images for each class in the dataset

2.2 Data Exploration

All images are of typical RGB color space. The dimensions of images vary, with the range of width is from *1,192 to 1,732*, and the range of height from *670 to 974*. There are 10 image shapes. Overall, the aspect ratio (width : height) of all the images falls into two categories: *16 : 9* and *4: 3*, see Fig. 2. Images in shape of *16 : 9* account for the vast majority of the datasets.

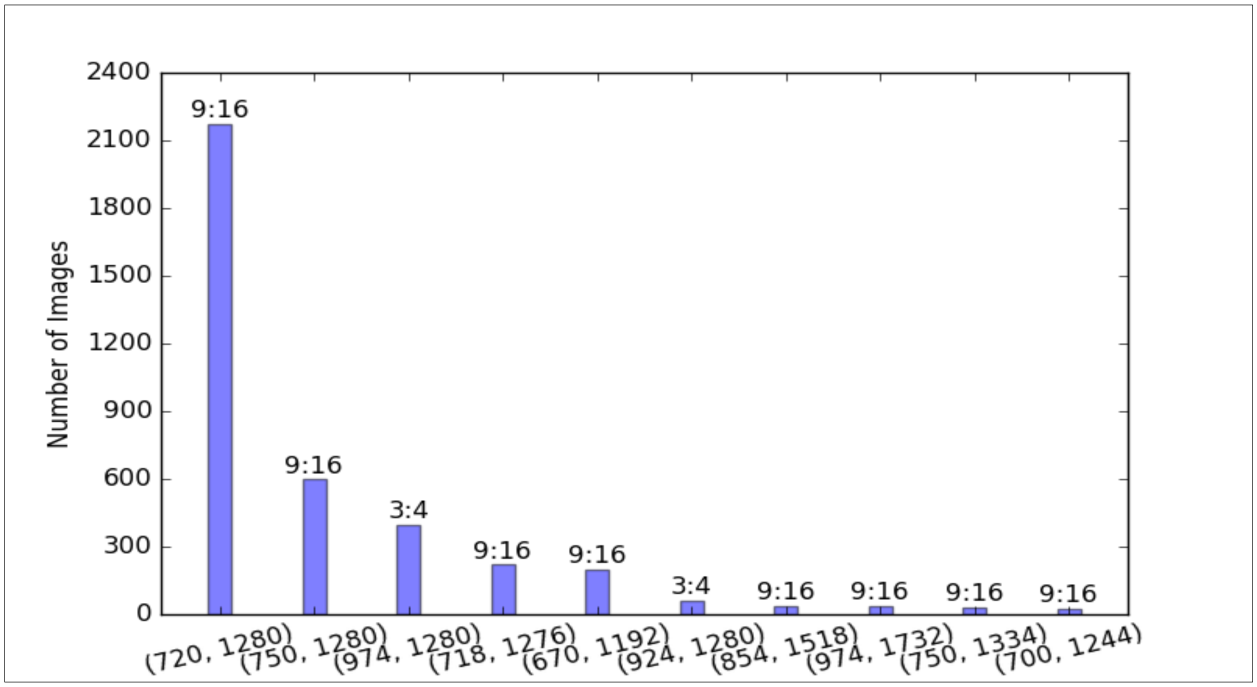


Fig. 2 Image size summary of the dataset

3. Methods

The problem to be solved is a typical image classification problem. To identify the objects from thousands of images, a model with the two following attributes should be adopted: (1) having a strong learning ability and (2) having prior information built-in the design [*Haykin*, 1994]. Therefore, a convolutional neural network (CNN) is proposed to solve it, specifically for the following reasons: (1) the width and depth of CNN control the learning ability of the network, and they can be adjusted properly based on the model performance. During the training, the model is learning from the domain knowledge, which is mapping images to corresponding labels. (2) the prior information for image classification is that humans are sensitive to spatial patterns embedded in the received information. The convolution operations in a CNN work as hierarchical feature extractors, and is powerful and robust at recognizing local 2-D patterns and achieving high accuracy with a small number of parameters [*LeCun et al.*, 1998]. Overall, CNN takes images as input, and processes them by extracting local spatial patterns as features, and integrates the information from the features to generate a probability distribution over the image classes for each single image. The image class with highest probability will be considered as the label for the input image, see Fig. 3. For example, in Fig. 3, CNN receives an image of shark, and generates a probability distribution over the class types in the dataset. If the probability of shark is highest, this input image is correctly identified. CNN is widely used in image classification and recognition [*Krizhevsky et al.*, 2012; *Szegedy et al.*, 2015; *Zeiler and Fergus*, 2014], and enjoys a great reputation. In light of the impactful success of the CNN framework, it is rational to believe that CNN could be a solution to the problem in this report.

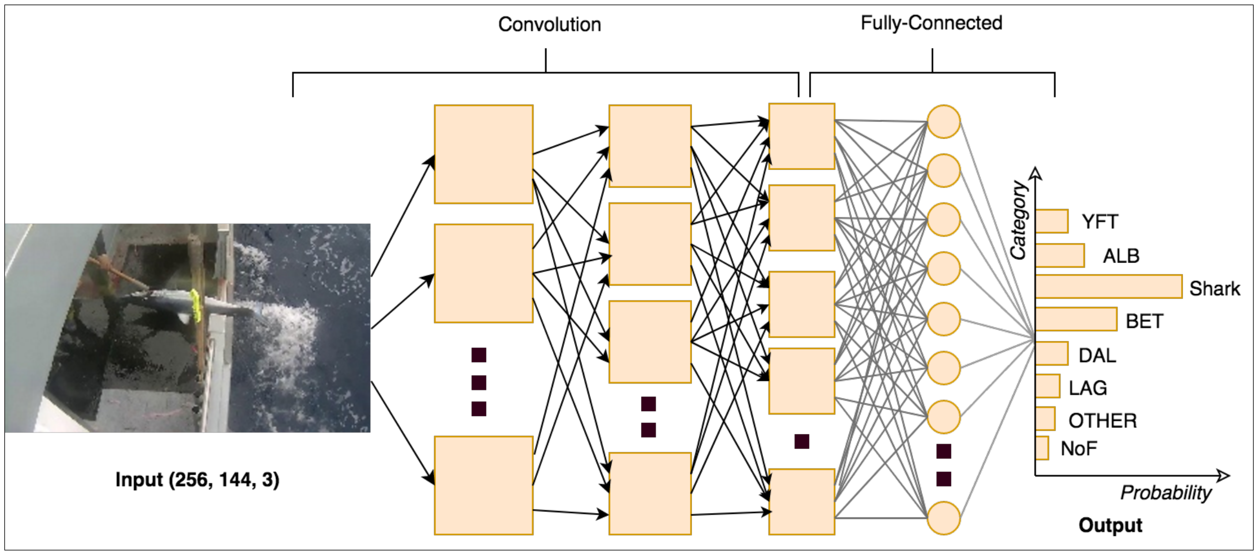


Fig. 3 Image recognition flow of CNN.

3.1 Data Preprocessing

Frist of all, the dataset has a serious class imbalance problem, as identified in section 2.1. To solve this problem, a data augmentation method is employed. The data augmentation is a series of processes to generate synthetic images based on the existing ones. The processes include two subprocesses: (1) randomly changing image hue, contrast, brightness and saturation without image deformation and (2) randomly deforming images through rotation, shifting, shearing and horizontal flipping. Also, the shape of raw images is in order of 10^3, which needs too much computing power to conduct the following model training task. Therefore, down-sampling the raw images is necessary. For keeping the aspect ratio (*16:9*) of the majority images, the targeted shape of new image size is *256 (width) : 144 (height)*. The image resizing is conducted through *OpenCV* package, and the subprocess (1) and (2) are completed through *tensorflow.image()* and *keras.preprocessing.image.ImageDataGenerator()* respectively. After image resizing and data augmentation, the class imbalance problem is mitigated, see Fig. 1. There are 14,993 images (including resized original images) in the dataset. Meanwhile, the eight image classes are encoded numerically, thus, numbers zero to seven represent all the classes.

Second of all, a further downsizing images is conducted on the base of resized and synthetic images. The further-downsized images are in shape of 64 *(width) : 36 (height)*. Both datasets in format of (246, 144, 3) and (64, 36, 3) are not preprocessed in any other way before training. While training, the dataset is imported into RAM and normalized by being divided by 255.

Finally, transfer learning is employed in the model refining process. Transfer learning is a method of reusing image features learned on other dataset by other models. Those learned features (also known as bottleneck features) usually can be used on new task to achieve high model performance. In this report, VGG16 is chosen to compute the bottleneck features, which were learned on ILSVRC-2012 dataset [*Simonyan and Zisserman*, 2014].

3.2 Benchmark Model

LeNet-5 is chosen as a benchmark model for this problem. LeNet-5 is a classical convolutional neural network that has been used for digit classification. LeNet-5 with artificially augmented training data have achieved accuracy of 99.2% at classifying digits [*LeCun et al.*, 1998]. LeNet-5 consists of eight layers, including input and output. Following the input layer, there is two consecutive convolutions and max-pooling layers, which accounts for the layer two to five. After that, there are two fully-connected layer accounting for layer six to seven, formed by each element of all feature maps in the previous max-pooling layer. For convolution layers, the activation function is chosen as 'ReLU', no padding is added. For max-pooling layers, the pool size and strides are both (2, 2). The layer seven connects to the output layer. The output layer has the same number of neurons as the number of classes. The activation function is chosen as 'Softmax'. For this problem, LeNet-5 architecture (width and depth) will be adopted as for benchmark model, with changes in input and output layer dimensions. The input dimension for each single image will be 16:9. The output layer will consist of 8 nodes to represent 8 classes. The LeNet-5 results will be measured by F1 score, which is introduced in section 3.4. The architecture for benchmark model is shown in Fig. 4 (a).

3.3 Model Refining

There are two strategies on model refining: (1) fine tuning LeNet-5 by adjusting hyper-parameters such as number of filters (width of layers), learning methods, learning rates and regularization techniques; and (2) building a model on top of established model structure, VGG16. There are two objectives of using two strategies: (1) to compare the model performances between fine-tuned LeNet-5 and transfer learning model, for exploring the factors affecting model performance; and (2) to discover a model with best predictability. The architecture for transfer learning model is shown in Fig. 4 (b). Using the bottleneck features computed from the top 13 convolutional layers as input, a fully convolutional (FCN) model is employed.

Tuning the transfer learning model is critical to improve model performance. An exhaustive grid method is employed to identify the best combination of parameters. F1 score on validation dataset is the criterion. In this project, a grid of parameters includes optimizers, weight initialization, epochs, and batch size. The optimizers include “sgd”, “rmsprop” and “adam”. The weight initialization methods include “glorot_uniform”, “glorot_normal”, and “uniform”. Epochs include 10, 20, and 30. Batch sizes include 32, 64 and 128. The parameter grid provides 81 combinations of parameters. In addition, cross-validation (4 fold) method is used while searching for the best parameters. Using a 3.4 GHz Intel Core i7 CPU, the model, with architecture “VGG16 +Top_ FCN (256-256-8)”, see Table 1, run 15 days and identified the best parameters for the transfer learning model, is ‘epochs’: 30, ‘batch_size’: 128, ‘optimizer’: ‘adam’, ‘initialization’: ‘glorot_normal’.

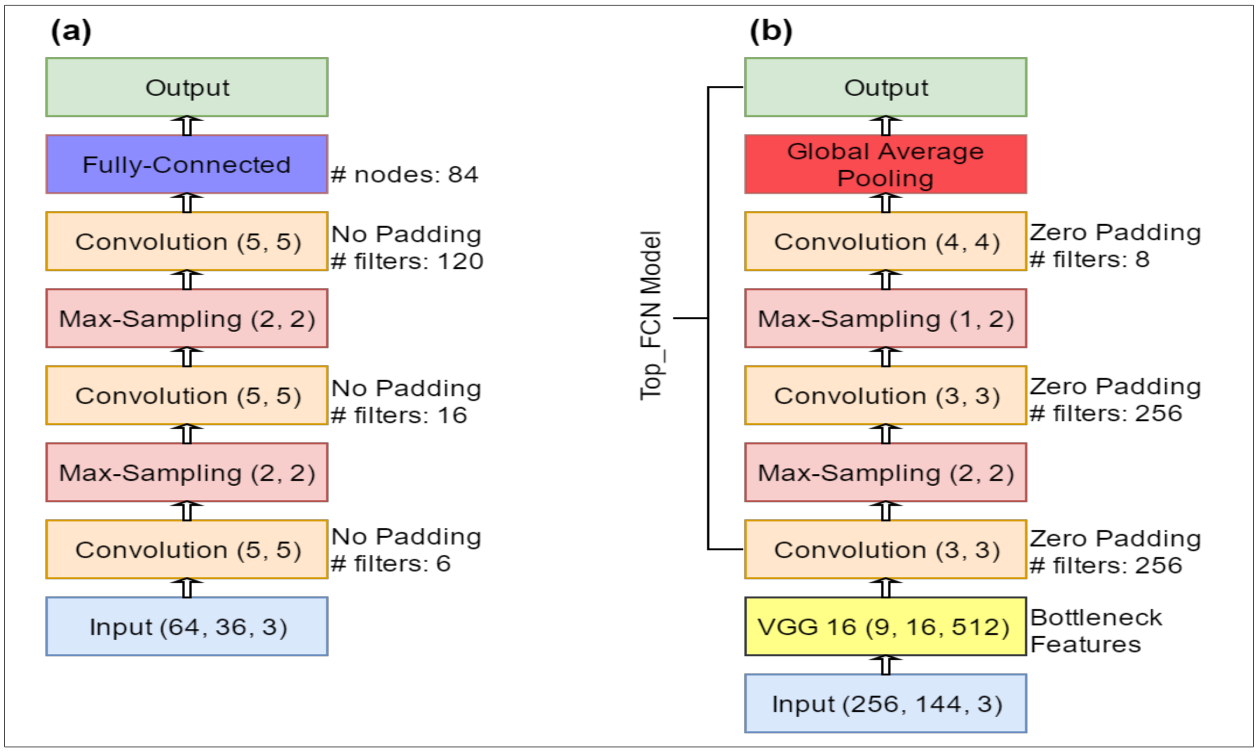


Fig. 4 Architectures of Standard LeNet-5 (a) and Transfer Learning Model (b).

3.4 Evaluation Metrics

F-score is commonly used to evaluate classifier performance while the class distribution is not perfectly uniformly distributed [*Ye et al.*, 2012]. Furthermore, F-score is a metric that trades off recall and precision [*Manning et al.*, 2009; *Rijsbergen*, 1974]. In this study, recall and precision are defined as

$$recall = \frac{fish\ types\ identified\ and\ correct}{total\ fish\ types\ correct} \tag{1}$$

$$precision = \frac{fish\ types\ identified\ and\ correct}{total\ fish\ types\ identified} \tag{2}$$

where "fish types identified" means the fish types that are predicted by the model. Intuitively, recall is about the percentage of all "fish types identified" (including correct and incorrect ones) that is correctly identified (retrieved) by the model. Precision is about the percentage of "fish types identified and correct" among all the predicted positives. While assigning same weights for both recall and precision, the harmonic average of recall and precision is F1, which is defined as [*Lipton et al.*, 2014]

$$F1 = \frac{2}{\frac{1}{reall} + \frac{1}{precision}} \tag{3}$$

This metric not only can tell how well the model performs overall in terms of predicting unseen samples, but also can give modelers insights on how model performs on each specific class. Thus, F1 score is proposed to evaluate the performance of image classification models.

3.5 Class Activation Map

To understand why the convolutional neural network is capable of learning to identify various images, it is necessary to investigate what it learned through the model training. A class activation map (CAM) for a single image suggests that discriminative local patch learned by the model to determine its category. In this study, a method using global average pooling on final convolutional layer in the CNNs is employed to generate CAMs [*Zhou et al.*, 2015]

4. Results

4.1 Model Evaluation and Validation

The results of the model performance on test dataset are summarized in Table 1. The standard LeNet-5 can achieve F1 score 0.5778 without using any regularization technique. While using batch normalization on convolutional layers, and dropout on the last fully-connect layer, the F1 score does not improve much. The zero-improvement with applying regularizations on standard LeNet-5 suggest that it may not have the capacity to learn necessary features to identify the various types of fish. Increasing the number of filters in convolutional layers is one common way to increase the model capacity. As shown in the Table 1, with increasing of the number of filters in all three convolutional layers, the F1 score on test dataset increases. With number of filters of 24-

64-96 in convolutional layers, the F1 score reaches up to 0.7353. However, setting the equal number of filters in first two convolutional layers (64-64-128), the model performance deteriorates to a great extent. The phenomenon may suggest that the second convolutional layer with equal number of filters as the first one may lose information learned in the first one. Therefore, the high hierarchical features in the second convolutional layer needs more feature maps to be captured than the first convolutional layer does. Furthermore, increasing of filter number in convolutional layers cannot improve the model performance, which suggests that the model capacity is full under the standard LeNet-5 architecture.

Transfer learning comes into play to boost the performance of the classifier. The outputs from convolutional layers of VGG16 provide learned features on the ILSVRC-2012 dataset. Using the bottleneck features as input of the fully convolutional (FCN) model, the F1 score increases to 0.8939. There are two reasons that why transfer learning works well: (1) It is original input is in format of (144, 256, 3), which is larger than the standard LeNet-5 input. Thus, it means there are more information for transfer learning model to learn; (2) The bottleneck features help. The bottleneck features are extracted from previously well-established model that was training on millions of images. Therefore, the bottleneck features are characterized with stronger generalizability than the raw images.

Table 1 Comparison of F1 score on test set. Standard LeNet-5 is in italic. *The (6-16-120) indicates that there are 6, 16 and 120 filters in the 1st, 2nd and 3rd convolutional layers respectively.*

Input	Model Architecture of Convolutional Layers	F1 Score	Regularization
(36, 64, 3)	*LeNet-5 (6-16-120)*	*0.5778*	*N/A*
(36, 64, 3)	LeNet-5 (6-16-120)	0.5881	Batch Normalization & Dropout (0.5)
(36, 64, 3)	LeNet-5 (16-32-64)	0.7015	Batch Normalization & Dropout (0.5)
(36, 64, 3)	**LeNet-5 (24-64-96)**	**0.7353**	**Batch Normalization & Dropout (0.5)**
(36, 64, 3)	LeNet-5 (32-64-128)	0.7345	Batch Normalization & Dropout (0.5)
(36, 64, 3)	LeNet-5 (64-64-128)	0.4733	Batch Normalization & Dropout (0.5)
(36, 64, 3)	LeNet-5 (64-128-256)	0.7255	Batch Normalization & Dropout (0.5)
(36, 64, 3)	LeNet-5 (256-256-256)	0.7123	Batch Normalization & Dropout (0.5)
(144, 256, 3)	VGG16 + Top_ FCN (24-64-8)	0.8470	Batch Normalization
(144, 256, 3)	**VGG16 + Top_ FCN (256-256-8)**	**0.8936**	**Batch Normalization**

4.2 Class Activation Maps (CAMs).

The transfer learning model “VGG16 + Top_FCN (256-256-8)” achieves the highest F1 score on test dataset, it is critical to know what the model has learned during the training that make it perform well on prediction. Fig. 5 shows a set of images with the activation maps on top. The region covered with warm purple color indicates the local pattern that ignites the final prediction of the model. For example, the image on the left in 2nd row is correctly identified as shark, the ignition of the region is where a shark is. Furthermore, the images on the left in 3rd and 4th row in Fig. 5 show they are DOL and ALB respectively, and the local ignition regions spot where the DOL and ALB are correctly.

In addition, the CAMs can also help researcher identify hidden problems. For example, on the right of the 1st row of Fig. 5, the image was correctly identified as LAG. However, the local region that ignites the final correct prediction focuses on a person in the image, instead of fish. This phenomenon suggests that there is data leakage in this dataset for class LAG, so that the model can make correct prediction using features unrelated to the fish type.

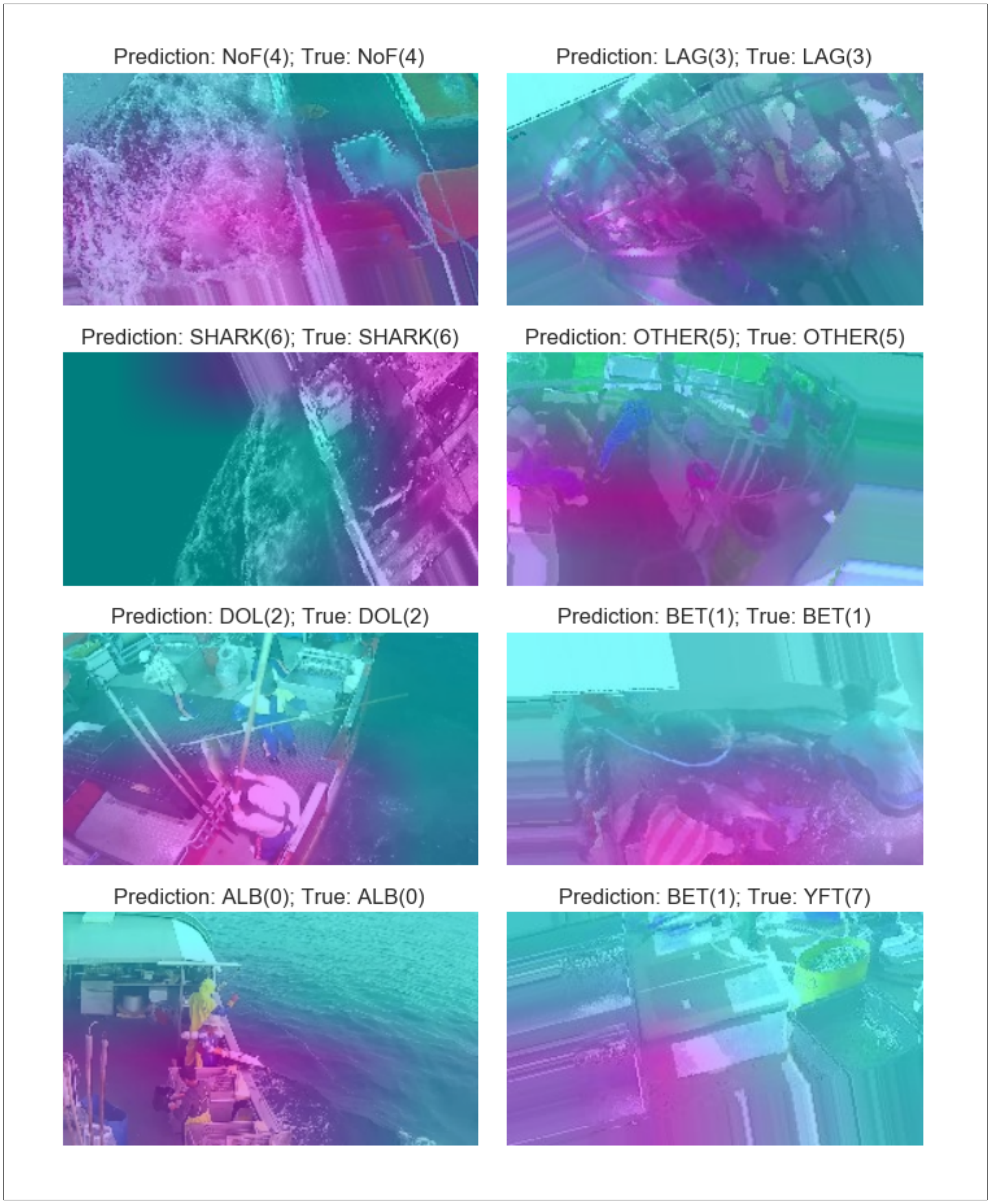

Prediction: NoF(4); True: NoF(4)
Prediction: LAG(3); True: LAG(3)
Prediction: SHARK(6); True: SHARK(6)
Prediction: OTHER(5); True: OTHER(5)
Prediction: DOL(2); True: DOL(2)
Prediction: BET(1); True: BET(1)
Prediction: ALB(0); True: ALB(0)
Prediction: BET(1); True: YFT(7)

Fig. 5 Examples of Class Activation Maps.

5. Conclusion

Convolutional neural networks with various architectures are explored in this report, to improve the model performance on identifying fish types. Transfer learning is considered to be the best method to build a classifier to take this task. The model "VGG16 + Top_FCN (256-256-8)" that achieves F1 score of 0.8936 is very good at recognizing ALB, DOL, LAG and SHARK, see Table 2. On the other hand, it still needs improvement on recognizing other four classes, especially on BET and YFT.

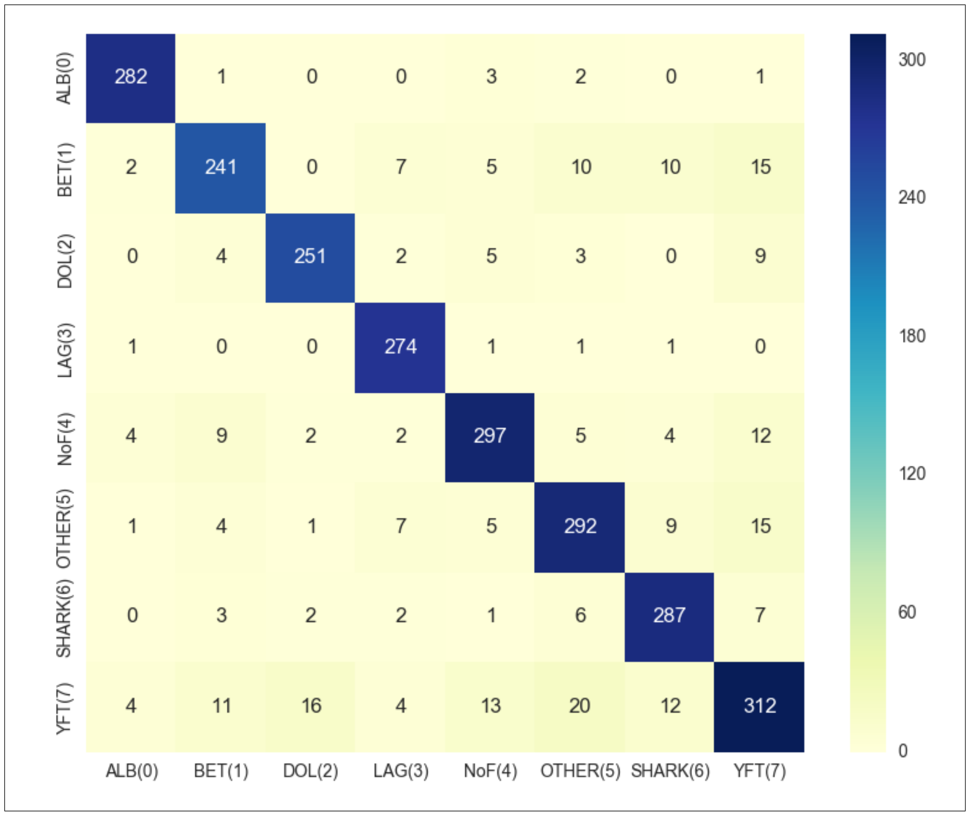
Table 2 Classification report of F1 score on test dataset

	Precision	Recall	F1 Score	Support
ALB(0)	0.96	0.98	**0.97**	289
BET(1)	0.88	0.83	0.86	290
DOL(2)	0.92	0.92	**0.92**	274
LAG(3)	0.92	0.99	**0.95**	278
NoF(4)	0.90	0.89	0.89	335
OTHER(5)	0.86	0.87	0.87	334
SHARK(6)	0.89	0.93	**0.91**	308
YFT(7)	0.84	0.80	0.82	392
weighted average	0.89	0.89	**0.8936**	2500

Furthermore, the confusion matrix on test dataset can identify where the model confuses the most while predicting the fish types. As shown in Fig. 6, the model misclassifies BET as YFT, SHARK and OTHER; and it misclassifies YFT as OTHER, DOL, NoF and SHARK frequently while making wrong predictions. It is believed that using more high quality data on class BET, YFT, SHARK, NoF and OTHER would be helpful for the model to learn more unique features for each lass. There are several ways to improve the model: (1) collecting more data for each class; (2) further tuning the model by carefully adjusting learning rate, weight decay rate etc; (3) exploring more model architectures; and (4) using other successful models such as VGG19 or ResNet50 [*He et al.*, 2015]to compute bottleneck features in transfer learning.

There are two challenging aspects about this project: (1) discovering the best model architecture to achieve high prediction performance. The width and depth of a CNN model have significant influence on model performance. Securing a successful model architecture is a process of trial and error, which requires enormous efforts; (2) Fine turning a model based on previous model score. For example, the model "VGG16 + Top_FCN (256-256-8)" works satisfactorily on identifying ALB; however, still has room for improvement on recognizing YFT, see Table 2. The two aspects are challenging for this project, and probably so for other similar image classification problem as well.

Overall, this project provides a great opportunity to explore how convolutional neural network works on image recognition and to apply learned features from different dataset on this new task. A well-trained model can be used on fishery monitoring to help fishermen and regulatory agencies build a more sustainable world.



	ALB(0)	BET(1)	DOL(2)	LAG(3)	NoF(4)	OTHER(5)	SHARK(6)	YFT(7)
ALB(0)	282	1	0	0	3	2	0	1
BET(1)	2	241	0	7	5	10	10	15
DOL(2)	0	4	251	2	5	3	0	9
LAG(3)	1	0	0	274	1	1	1	0
NoF(4)	4	9	2	2	297	5	4	12
OTHER(5)	1	4	1	7	5	292	9	15
SHARK(6)	0	3	2	2	1	6	287	7
YFT(7)	4	11	16	4	13	20	12	312

Fig. 6 Heat map of confusion matrix on test dataset.

References:

Aqorau, T. (2001), Tuna Fisheries Management in the Western and Central Pacific Ocean: A Critical Analysis of the Convention for the Conservation and Management of Highly Migratory Fish Stocks in the Western and Central Pacific Ocean and Its Implications for the Pacific Island States, *The International Journal of Marine and Coastal Law*, *16*(3), 379-431.

FAO (2016), The State of World Fisheries and Aquaculture, edited by FAO, p. 200, Rome.

Haykin, S. (1994), *Neural Networks : A Comprehensive Foundation*, 1st ed., 696 pp., Macmillan College Publishing.

He, K., X. Zhang, S. Ren, and J. Sun (2015), Deep Residual Learning for Image Recognition, *CoRR*, *abs/1512.03385*.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012), ImageNet Classification with Deep Convolutional Neural Networks, paper presented at Neural Information Processing Systems 2012.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998), Gradient-Based Learning Applied to Document Recognition, paper presented at Proceedings of the IEEE,.

Lipton, Z. C., C. Elkan, and B. Naryanaswamy (2014), Thresholding Classiers to Maximize F1 Score, *Mach Learn Knowl Discov Databases*, 225-239, doi:10.1007/978-3-662-44851-9_15.

Manning, C. D., P. Raghavan, and H. Schütze (2009), Evaluation in information retrieval, in *An introduction to information retrieval*, edited.

NIC (2016), Global Implications of Illegal, Unreported, and Unregulated (IUU) Fishing, edited by NIC.

Rijsbergen, C. J. v. (1974), Foundation of Evaluation, *Journal of Documentation*, *30*, 365-373.

Simonyan, K., and A. Zisserman (2014), Very Deep Convolutional Networks for Large-Scale Image Recognition, *CoRR*, *abs/1409.1556*.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015), Going Deeper with Convolutions, paper presented at The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston.

Ye, N., K. M. A. Chai, W. S. Lee, and H. L. Chieu (2012), Optimizing F-Measures: A Tale of Two Approaches, paper presented at the 29th International Conference on Machine Learning, Edinburgh, Scotland, UK,.

Zeiler, M. D., and R. Fergus (2014), Visualizing and Understanding Convolutional Networks, *Computer Vision - Eccv 2014, Pt I*, *8689*, 818-833.

Zhou, B., A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba (2015), Learning Deep Features for Discriminative Localization, *CoRR*, *abs/1512.04150*.