Application of AIOT on fishery industry

專題生：吳承威，王樸

指導教授：林政宏

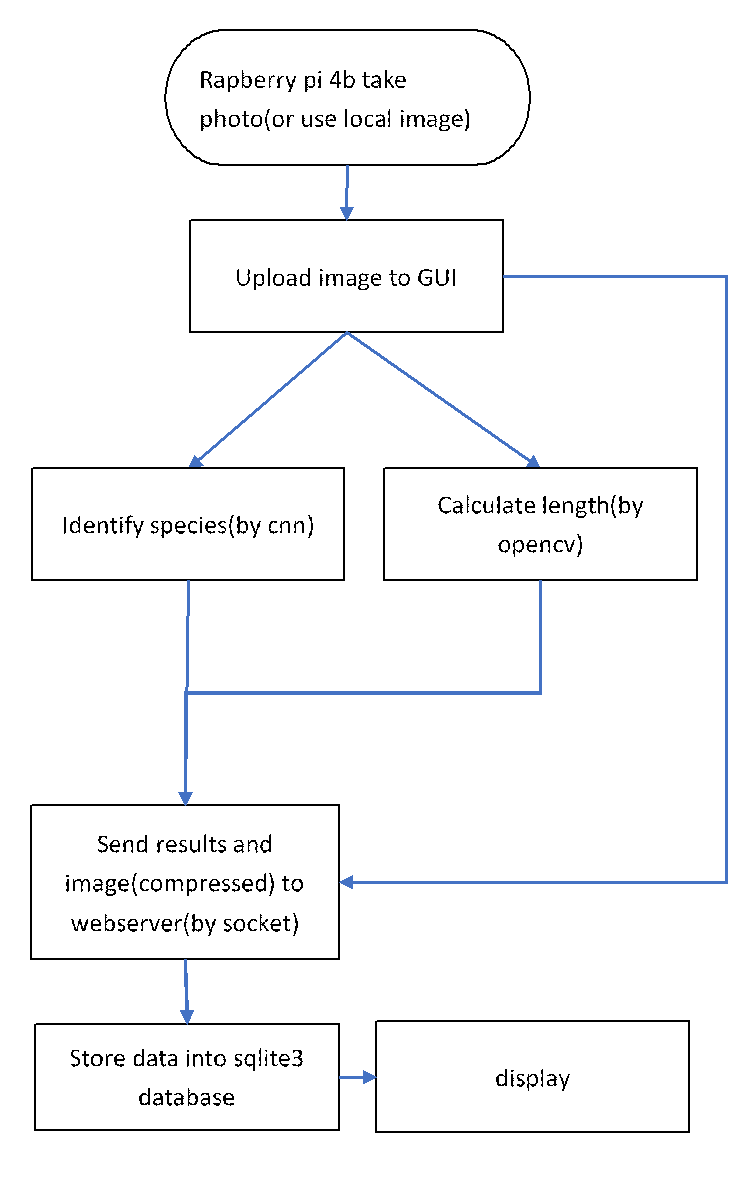
[[1]](#footnote-1) *摘要*—Artificial Intelligence and Internet of Things are booming in recent years. Our project is the application of both of above with image processing technology.

# 導論

Fishery has always been an important industry in Taiwan. However, it is difficult to identify the fish species, calculate the length and record it in time when a fish is caught. Therefore, we designed a device to solve the problem. We operate neural network on the Raspberry Pi 4b to detect fish species, and use python Open CV to capture the frame so that we can calculate the fish length. Then we design a GUI to input fish photo and display the results on a web page.

# 原理與架構簡介

## Structure and device





We use the Raspberry Pi 4b connected with the stick and the monopod as a device.

Raspberry Pi 4b connected to the screen, so we can upload the picture taken on site or local images in the machine to GUI. Then, we can identify fish species and calculate fish length. Finally, we can send data to webserver.

## Calculating the length

For this use very simple methods of image processing and math to calculate the length of the fish. It consists of getting the contour of the fish using Otsu threshold and a method from Open CV called minarearect(), to calculate the length in term pixel of the fish.



Also to calculate the length per pixel , we use the equation:

Where N is the number of pixels in this axis and y is the distance between camera and the object.

Once we have the length in term of pixel, we can calculate the real length in centimeter by multiplying the length per every pixel we calculated before.

where x is actual length

## Model architecture

For the classification part of the program we use neural networks for this job. Originally we decided to use the model YOLO v4 for our project, but because the model was too big for a raspberry pi 4 to run properly, with high accuracy, we decided to discard the model.

So the model we decided to use is inspired on the model that appears on the paper [1]. Which better suit our necessity of lightness and accuracy. The difference is that our model has an extra dense layer before the output. The reason of this, is to give the network an extra layer of liberty.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| layer | number of kernels | size | stride | padding | activation | dropout |
| Inputs | 400x400x3 | | | | | |
| conv1 | 32 | 5x5 | 1x1 | 2x2 | ReLU |  |
| maxpool1 |  | 5x5 | 5x5 | 1x1 |  |  |
| conv2 | 64 | 5x5 | 1x1 | 2x2 | ReLU |  |
| maxpool2 |  | 5x5 | 5x5 | 1x1 |  |  |
| conv3 | 32 | 5x5 | 1x1 | 2x2 | ReLU |  |
| maxpool3 |  | 5x5 | 5x5 | 1x1 |  |  |
| Flatten | 288 | | | | | |
| Dense1 |  | 2048 |  |  | ReLU | 0.2% |
| Dense2 |  | 2048 |  |  | ReLU | 0.2% |
| Dense3 |  | 5 |  |  | softmax |  |

|  |
| --- |
| D:\desktop\AIoT\cnn_graph\lossplot.png |

# 實驗步驟、過程、與結果

## Preprocessing

Before feeding the dataset to the neural network, we make some extra processing to artificially increase the data set, to make sure that the model doesn’t over feat. This processes are: image random rotation, random horizontal or vertical flip, add random perspective and random image crop.

Also, because the model we use need specific image size (400x400x3), we need to resize the image to meet this condition. We want to feed the network with the same aspect ratio, and we don’t want that the image of the fishes gets distorted, so if the image does not have the same image ratio we add black spaces, to equalize the aspect ratio and then resize.



## Datasets

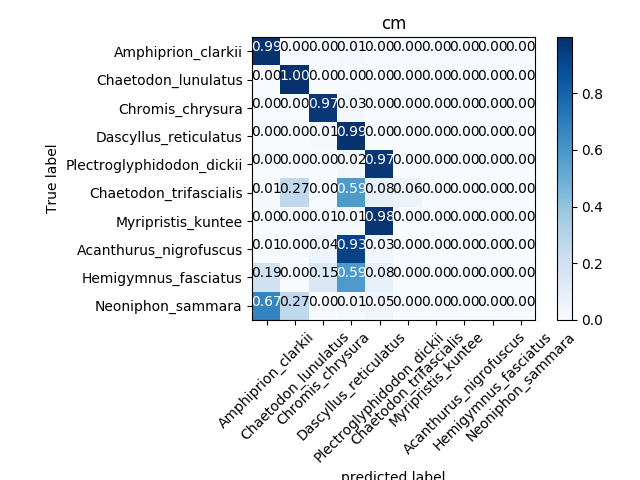
### Test model dataset

We use this dataset

<http://groups.inf.ed.ac.uk/f4k/GROUNDTRUTH/RECOG/>

and select 10 classes which have more data from it to train our neural network. In this case the first five classes have more than 2000 images, specially the first one which has 11312. Summing this five classes number of images it makes more than 18000 images. However, the last 5 classes have less than 200 images per class.

As difference of amount of data, we can see the difference of result from the confusion matrix below. The first 5 classes have been identified correctly with accuracy more than 97%. However,the last 5 classes either have been misidentified or have low accuracy.



### Our dataset

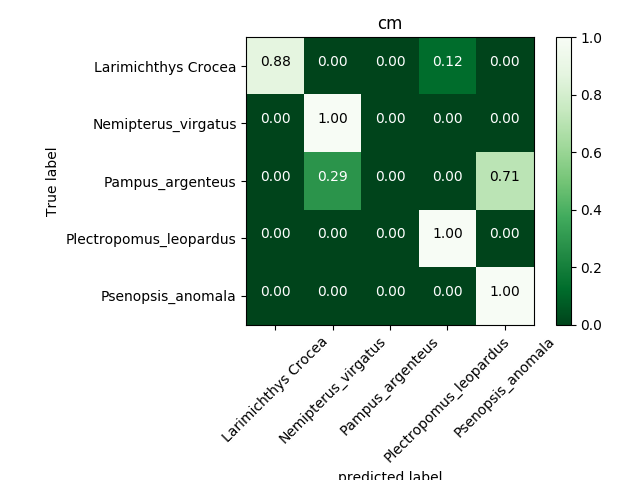
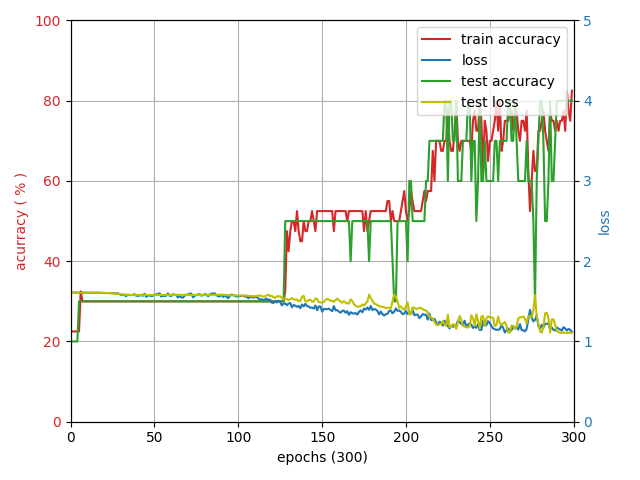
#### We couldn’t find proper fish dataset to meet our project’s needs, and it is difficult to collect data in fishing harbor, we decide to collect data in Taiwanese markets.

The dataset consists of 5 classes of fishes, that are common in Taiwanese fish markets. This are: Larimichthys Crocea (a), Nemipterus virgatus (b), Pampus argenteus (c), Plectropomus leopardus (d), Psenopsis anomala (e). This classes are selected based on the amount of images we collected.

 (a)  (b)  (c)  (d)

 (e)

Originally the data set consist of photos that we took directly from the market, plus some other images downloaded for the internet. Due to the lack of image, which does not exceed 30 photos per class, the best accuracy that we managed to get does not get over 80%.



From the confusion matrix of the model we can see that the classes that has more trouble classifying are Pampus argenteus (a) and Psenopsis anomala (b).

(a)  (b)

As it can be seen, both fishes have a very similar shape, color and background, which indicates that the data we have is not enough. So we decided that we have to do some data augmentation, manually.

## Manual data augmentation

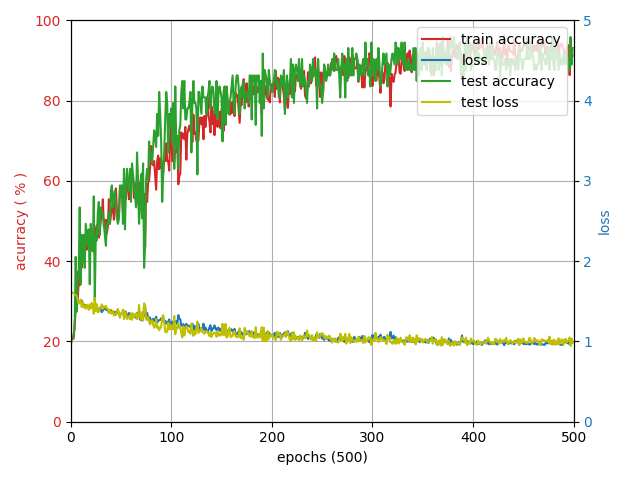
The methods we use to add extra data consist, using a photo editor and change the background of the some of the fishes, a rotation with respect of the original photo, and changing the position of it.

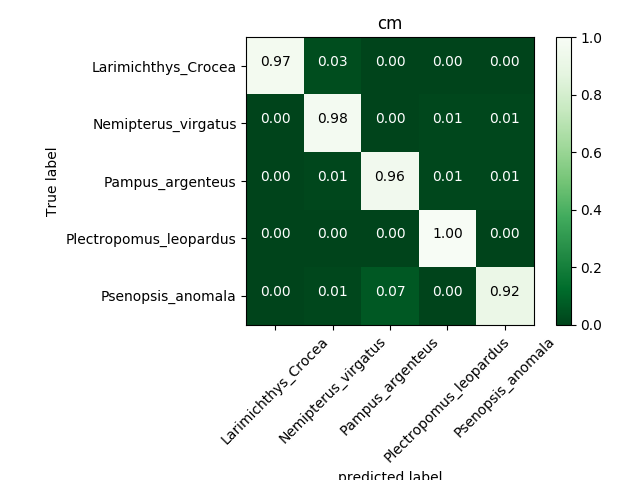
The goal of this method is make sure that the network can differentiate the fish from the background, and the fact that not all the fishes of the same class has the same shape, position, rotation and other small details.



This was applied on 2 of the fishes of every class. And with different backgrounds increasing from less than 30 photos per class to more than 70 photos per class.

By doing this we manage to increase from the original 80 % of accuracy to 96 % of accuracy.

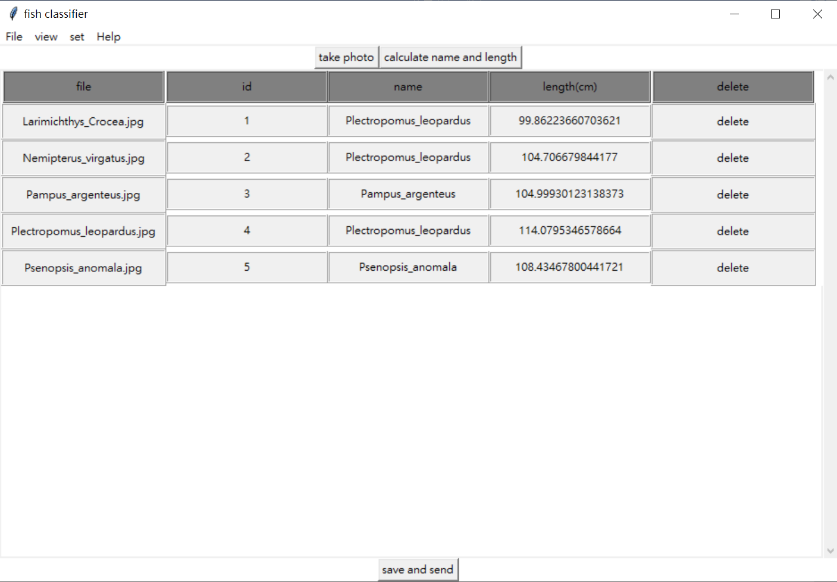




From the confusion matrix it can be seen that the manual data augmentation has an impact on the training results.

## GUI and Webpage

(1) GUI: We can take photo or send local data to GUI. Then we can identify the species and calculate fish length. Finally, we can send results to web server database and display.



(2) Webpage: We use webpage with select function by name or length to display the results from database.



# 結論

Although our identification accuracy more than 90% in all five species, the data set still too small that our results sometimes go wrong. On the other hand, data augmentation when training neural network with few data. We will continue to make the network more robust, by adding new data and using new data argumentation methods.

# 誌謝

Thank for help of seniors in lab and professor.

參考資料

[1] Rathi, D., Jain, S., and Indu, S., “Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning”, https://arxiv.org/abs/1805.10106v1, 2018.

[2] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR (2016)

[3] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, pages 6517–6525. IEEE, 2017.

[4] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. CoRR, abs/1804.02767, 2018.

[5] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao , “Yolov4: Optimal speed and accuracy of object detection,” arXiv preprint arXiv:2004.10934, 2020.

1. [↑](#footnote-ref-1)