1.0 PROBLEM STATEMENT

The fish identification challenge encountered during the study of ecological fish habitat has significantly limited the effectiveness of these studies. Some of the causes of this challenge are the use underwater cameras making manual recognition hard. Also, since water bodies are not always clear, this cameras produce poor visibility, have varying illuminations and capture many noise. Hence an efficient fish identification system was required.

This will consequently improve ecological fish habitat studies. A model was built which could effectively classify a random inputted images of a fish into 23 classes namely:

01.Dascyllus reticulatus, 02.Plectroglyphidodon dickii, 03.Chromis chrysura, 04.Amphiprion clarkia, 05.Chaetodon lunulatus, 06.Chaetodon trifascialis, 07.Myripristis kuntee, 08.Acanthurus nigrofuscus, 09.Hemigymnus fasciatus, 10.Neoniphon sammara, 11.Abudefduf vaigiensis, 12.Canthigaster valentine, 13.Pomacentrus moluccensis, 14.Zebrasoma scopas, 15.Hemigymnus melapterus, 16.Lutjanus fulvus, 17.Scolopsis bilineata, 18.Scaridae, 19.Pempheris vanicolensis, 20.Zanclus cornutus, 21.Neoglyphidodon nigroris, 22.Balistapus, 23.Siganus fuscescens.

2.0 DATA MANIPULATION

First, I mounted the Google drive and created a folder called Runmila for the purpose of this project and unzipped the dataset into the created folder.

In this project python modules such as CV2, Matplotlib, Keras, OS and Numpy Ire imported and used. CV2 was used for image augmentation and preprocessing, Matplotlib for visualization, Keras for Model definition, OS for directory path and Numpy for efficient numerical computation.

CV2 was used to view random samples of the dataset, I did a comparison with regards to the image sizes as I noticed some of the images Ire of different sizes, after the comparison and other deliberation I realized that I need some image preprocessing to be done on the datasets to enable the model work better with the dataset, the image preprocessing I did include the following:

Standardization: made sure images have the same pixel size

Reshaping: after manual scrutiny I made sure the fish classes are of the same shape thereby normalizing the dataset across all classes.

Data was separated into classes. For classes with small image sets I applied **flipping, rotating and skewing image** processing techniques to duplicate the pre-existing images and added to the same class thereby giving the model more images data set to work with. Classes with data below 200 images Ire increased with 135 using 15 random images from each classes.

The above image preprocessing done was to make the model work better and improve the overall accuracy of the model. If this was not done, there would be a great inherent bias by the model towards classes that had very large quantity of images.

After the preprocessing I printed the new number of images in the classes to show the added processed images in the classes where changes Ire made.

Then labels Ire created for the classes and the individual classes combined to a single dataset and **normalized**.

3.0 MODEL

A few models Ire tried out, namely: CNN with 1 Convolutional Layer, CNN with 2 Convolutional Layer and CNN with 3 Convolutional Layer.

The Convolutional Neural Network (CNN) model with 2 Convolutional layer was selected. The criteria considered was the test accuracy.

For Model exploration I started by splitting the processed dataset into train and test dataset with the training dataset taking 70% of the processed dataset

When defining the model, some parameters used are listed below

batch_size = 25

nb_classes =23 Number of classes

nb_epochs = 5 number of epochs

nb_filters = 32 number of filters

nb_pool = 2 number of pools

nb_conv = 3 number of convolutions

input_shape = (100,100,3)

activation function = ReLU and SoftMax

I made some test-run with other values and settled with these values as they provide the best accuracy after testing other values.

After training and fitting the datasets with the model, I plotted a history of loss graph to visualize the loss along with epochs, the training loss reduced from 0.6 to below 0.2, the test loss moved from 0.3 to 0.25 with 5 iterations. Model was saved for subsequent evaluation. The validation datasets Ire also saved.

Part B

As a new notebook, drive was mounted again for usage and the desired modules imported.

The saved model was loaded and the summary printed. The model summary summarizes the output shape of each layer, e.g. the shape of the resulting feature maps: the total number of lights per layer.

The model was then evaluated directly using the validation dataset. This shold that a final accuracy of 95.78% was achieved.

To understand the convolution of the CNN better. Filters and feature maps in the convoluted layers Ire visualized.

The filters detect **patterns** such as edges in an image by detecting the changes in intensity values of the image. The dark squares indicate small lights and the light squares represent large lights. To see all channels in a row for all 32 filters would require many subplots in which it may be challenging to see any detail. Therefore, it was limited to the first 8.

The feature maps capture the result of applying the filters to an input image. Where a random image from the class 22 was chosen as a sample.

At each layer, the feature map is the output of that layer. This shows what features our CNN detects.

A subset of the full model that would be the output of a given convolutional layer, to see the feature map was extracted and used for the process. All 32 maps Ire then plotted.

4.0 CONCLUSION

In this project, a model was built to classify images of fishes into 23 classes. The CNN based model created returned an accuracy over 95%, which signifies a very good model. This model is recommended to fish ecologist to aid their of ecological fish habitat studies by automating the fish identification process.