UNIVERSITI TEKNOLOGI MARA

UNIVERSITI TEKNOLOGI MARA (UiTM) SHAH ALAM

COLLEGE OF COMPUTING INFORMATICS AND MEDIA

BACHELOR OF COMPUTER SCIENCE (HONS.) (CDCS230)

ARTIFICIAL INTELLIGENCE ALGORITHMS (CSC583)

PROGRAM NAME

BIRD SPECIES CLASSIFICATION

PREPARED FOR

DR. NORZILAH BINTI MUSA

PREPARED BY

NAME	STUDENT ID	GROUP
NURADAM DANIAL BIN NOORMAN	2023125859	CS2303A
RAIMI IKHWAN BIN ROSLIN	2023389721	CS2303A
MUHAMMAD JAZLAN BIN SUKMAN SUZZAK	2023149761	CS2303A
MUHAMMAD HAZIQ BIN HASROL ALIAS	2023388537	CS2303A

DATE OF SUBMISSION

18 JULY 2023

TABLE OF CONTENTS

1.1 DESCRIPTION OF PROBLEM AND OBJECTIVE

1.1 PROBLEM STATEMENT

Bird species identification is a challenging task, both for humans and for computational systems. There are over 10,000 known species of birds, and many of them look very similar. This can make it difficult to identify a bird species from a single image.

There are a number of factors that make bird species identification difficult. First, birds can vary in appearance depending on their sex, age, and plumage. For example, male and female birds of the same species often have different colors. Second, birds can be difficult to photograph, especially in the wild. This can lead to blurry or low-quality images that make identification more difficult.

In recent years, there has been a growing interest in using artificial intelligence (AI) to solve the problem of bird species identification. AI-based systems can be trained on large datasets of bird images, and they can learn to identify bird species with high accuracy.

However, there are a number of challenges that need to be addressed before AI-based systems can be widely used for bird species identification. One challenge is the need for large and high-quality datasets of bird images. These datasets can be difficult and expensive to collect. Another challenge is the need for AI systems that are robust to variations in lighting, pose, and other factors that can affect the appearance of birds in images.

Despite these challenges, there is a growing optimism that AI can be used to solve the problem of bird species identification. AI-based systems have the potential to make bird species identification more accurate, efficient, and accessible. This could have a number of benefits, including helping to protect endangered species, improving our understanding of bird migration patterns, and making bird watching more enjoyable.

1.2 OBJECTIVES OF STUDY

The objectives of study for this project are:

- To develop a CNN-based model that can accurately classify bird species from images. This model will be trained on a large dataset of bird images, and it will be able to learn to identify bird species with high accuracy.
- To investigate the factors that affect the accuracy of CNN-based bird species classification models. These factors may include the size and quality of the training dataset, the architecture of the CNN model, and the optimization algorithm used to train the model.
- To evaluate the performance of the CNN-based model on a variety of bird species. This will help to determine the overall accuracy of the model and its ability to identify different bird species.

2.0 TECHNIQUE USED

Convolutional Neural Networks (CNNs) are a type of artificial neural network that is particularly well-suited for image recognition and classification tasks, making them a popular choice for tasks such as bird species classification. CNNs have revolutionized the field of computer vision and have enabled significant advancements in various applications, including self-driving cars, facial recognition, and medical imaging.

The basic idea behind CNNs is to mimic the visual processing that occurs in the human brain. Like the human visual system, CNNs consist of multiple layers of interconnected neurons that process information hierarchically. Each layer performs a specific function, gradually extracting more complex features from the input data. The following are three sorts of layers for feature extractions:

1. Convolutional layer

   Convolutional layers are the heart of CNNs. They apply a set of learnable filters (kernels) to the input image or feature map. Each filter detects a specific feature, such as edges, corners, or textures, by performing a mathematical operation called convolution. Convolution involves sliding the filter over the input and computing the element-wise multiplication between the filter weights and the corresponding pixels of the input. The results are summed up to produce a feature map that highlights the presence of the detected feature. Multiple filters are used to capture different features simultaneously. The parameters (weights) of these filters are learned during the training process.

2. Rectified Linear Unit (ReLU)

   ReLU is an activation function commonly used in CNNs. It introduces non-linearity to the network, allowing it to learn complex patterns and relationships. The ReLU function applies the following transformation: for any input value x, ReLU outputs max(0, x). In other words, if the input is negative, it is set to zero, and if it is positive, it remains unchanged. ReLU is preferred over other activation functions because it is computationally efficient and helps alleviate the vanishing gradient problem, which can hinder the learning process.

3. Pooling layer

   Pooling layers are used to reduce the spatial dimensions of the feature maps while retaining the most important information. The most common type of pooling is max pooling. It divides the feature map into non-overlapping regions (e.g., 2x2 or 3x3) and selects the maximum value within each region. The result is a downsampled feature map with reduced spatial resolution. Pooling helps achieve translation invariance by capturing the most salient features and discarding minor spatial variations. It also reduces the computational requirements of the network by decreasing the number of parameters and the overall size of the feature maps.

By combining convolutional layers, ReLU activation, and pooling layers, CNNs can effectively extract hierarchical features from the input images. The initial layers capture low-level features such as edges and textures, while the subsequent layers learn more complex features based on the previously extracted information. This hierarchical representation enables the network to understand the visual patterns and make accurate predictions for tasks like image classification, including bird species classification. These three layers work together to transform raw pixel values into meaningful representations that facilitate the learning and decision-making processes of the CNN.

3.0 INPUT, PROCESS, AND OUTPUT OF THE SYSTEM

3.1 INPUT, PROCESS, AND OUTPUT FOR EXPERIMENT

3.1.1 INPUT FOR EXPERIMENT

We built the image classifier for this project using hundreds of photos obtained online. These pictures were uploaded to our local storage and placed in different folders based on the class. We have 560 datasets in total. Therefore, we divided the data into two halves, with 420 or 75% used to train the model and 140 or 25% used for testing. This splitting is used to test whether or not our model is working on this issue. We will train our model with a training dataset and then compare the results to the accuracy rate. If the range between the two values is large, our model is probably overfitting.

```
Found 560 files belonging to 5 classes.
Using 420 files for training.
Found 560 files belonging to 5 classes.
Using 140 files for validation.
```

Figure 2.1 shows the datasets

There are also 5 bird categories: Broadbill, Eagle, Owl, Parrot and Woodpecker. We have 112 Broadbill dataset, 112 Eagle dataset, 112 Owl dataset, 112 Parrot dataset, and 112 Woodpecker dataset.

```
class_names = train_ds.class_names
print(class_names)
[2]  ✓ 0.0s
...  ['broadbill', 'eagle', 'owl', 'parrot', 'woodpecker']
```

Figure 2.2 shows the class names attributes on the datasets.

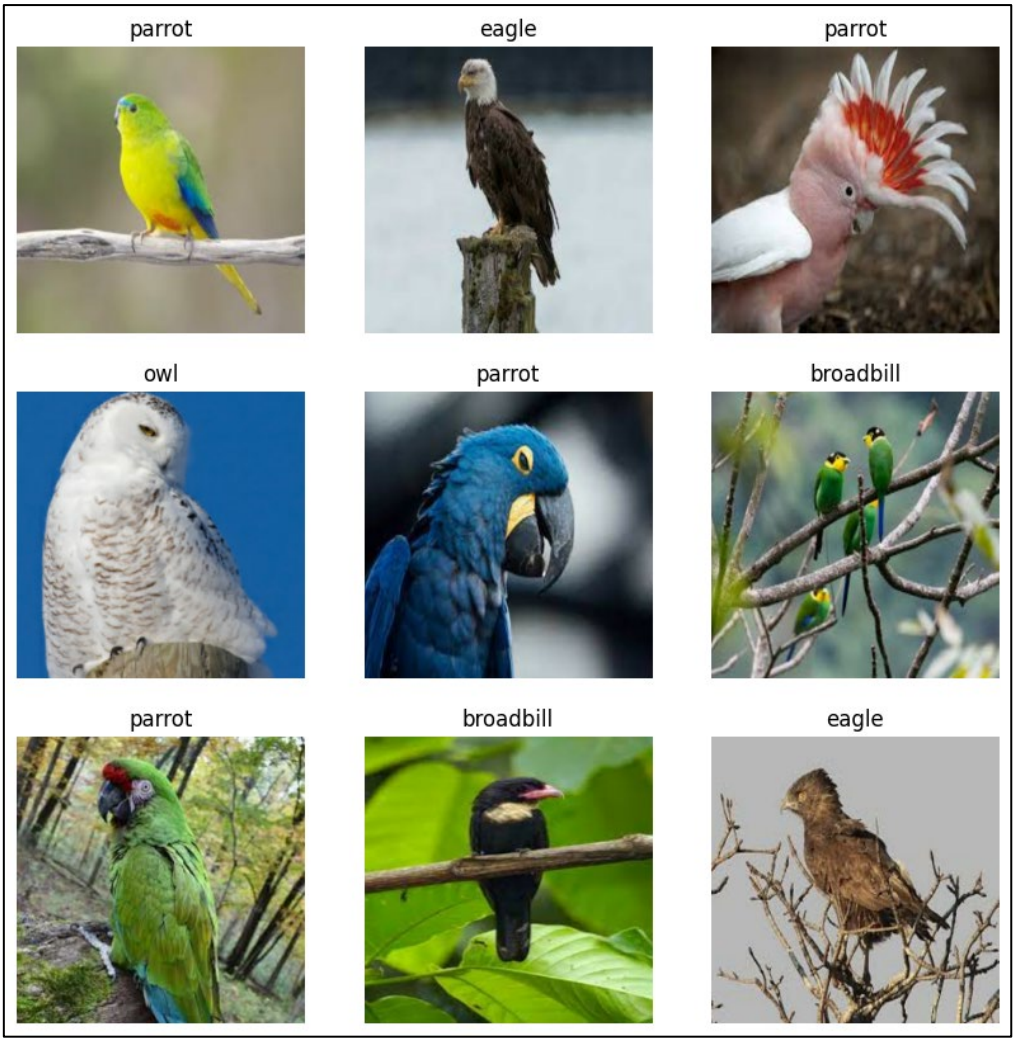

Figure 2.3 shows the first nine (9) images of training datasets

3.1.2 PROCESS FOR EXPERIMENT

Following completing the dataset collection, we begin the procedure by standardizing the data. That's also due to the RGB channel values being between [0,255]. This is not the ideal scenario for a neural network. Small input values are required. In this case, we normalize the values to be in the [0,1] range.

```
normalization_layer = layers.experimental.preprocessing.Rescaling(1./255)
```

Figure 3.1 shows the process of layer resizing

The model is therefore trained and created. The model is composed of three convolution blocks, each having a max pool layer. A ReLu activation function activates a completely linked layer with 128 units on top of it. In the initial learning, we instruct the machine to learn a total of 16 filters, and we set the kernel size to 3 × 3. The term "same" was also used as a placeholder for the padding parameter. Keras Conv2D parameters are split into two categories "same" and "valid." After that, maximum pooling is used.

```
num_classes = 5

model = Sequential([
  layers.experimental.preprocessing.Rescaling(1./255, input_shape=(img_height, img_width, 3)),
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])
```

Figure 3.2 shows how to create a model

For the second learning, we instruct the system to learn a total of 32 filters. The only thing we modify is the filter's total. The kernel size and padding parameters are the same as in the first learning. The computer is then commanded to learn a total of 64 filters for the third learning and 128 filters for the final learning. In short, the new learning employs twice the amount of filters as the prior one. The spatial volume decreases with each learning. The layers were therefore compiled using an optimizer, loss function, and metrics.

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Figure 3.3 shows the compiling model

Begin training by using the model fit method and setting the epoch to 20, indicating that there will be 20 passes over the whole training datasets.

```
epochs=20
history = model.fit(train_ds,validation_data=val_ds,epochs=epochs)
```

Figure 3.4 shows the setting of the epoch model

3.1.3 OUTPUT FOR EXPERIMENT

After data augmentation and machine learning, the result is obtained. Data augmentation is a sort of data analysis in which the angle, color, or other aspects of current data are changed.

Figure 3.5 shows the data augmentation

Then, use a model summary to inspect and document all of the network's levels.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 sequential_1 (Sequential)   (None, 300, 300, 3)       0

 rescaling_2 (Rescaling)     (None, 300, 300, 3)       0

 conv2d_3 (Conv2D)           (None, 300, 300, 16)      448

 max_pooling2d_3 (MaxPooling  (None, 150, 150, 16)     0
 2D)

 conv2d_4 (Conv2D)           (None, 150, 150, 32)      4640

 max_pooling2d_4 (MaxPooling  (None, 75, 75, 32)       0
 2D)

 conv2d_5 (Conv2D)           (None, 75, 75, 64)        18496

 max_pooling2d_5 (MaxPooling  (None, 37, 37, 64)       0
 2D)

 dropout (Dropout)           (None, 37, 37, 64)        0

...
Total params: 11,239,205
Trainable params: 11,239,205
Non-trainable params: 0
```

Figure 3.6 shows the output of the training model

After the training is done, we can see the accuracy rate. The percent of confidence for our project is 86.97. That means our images are strong and the images most likely belong to the categories.

```
1/1 [==============================] - 0s 207ms/step
This image most likely belongs to woodpecker with a 86.97 percent confidence.
```

Figure 3.7 shows the percentage of confidence

3.2 INPUT AND OUTPUT FOR USER INTERFACE

3.1.2 INPUT FOR USER INTERFACE

These are the predicted Graphical User Interface outputs below. We gathered random photographs from Google and organized them into a single folder. This application can detect 5 specieses of items which are broadbill, eagle, owl, parrot and woodpecker. The downloaded pictures are all based on the class that was supplied. The program for this UI will start by uploading the images, choosing the image and inserting the images.

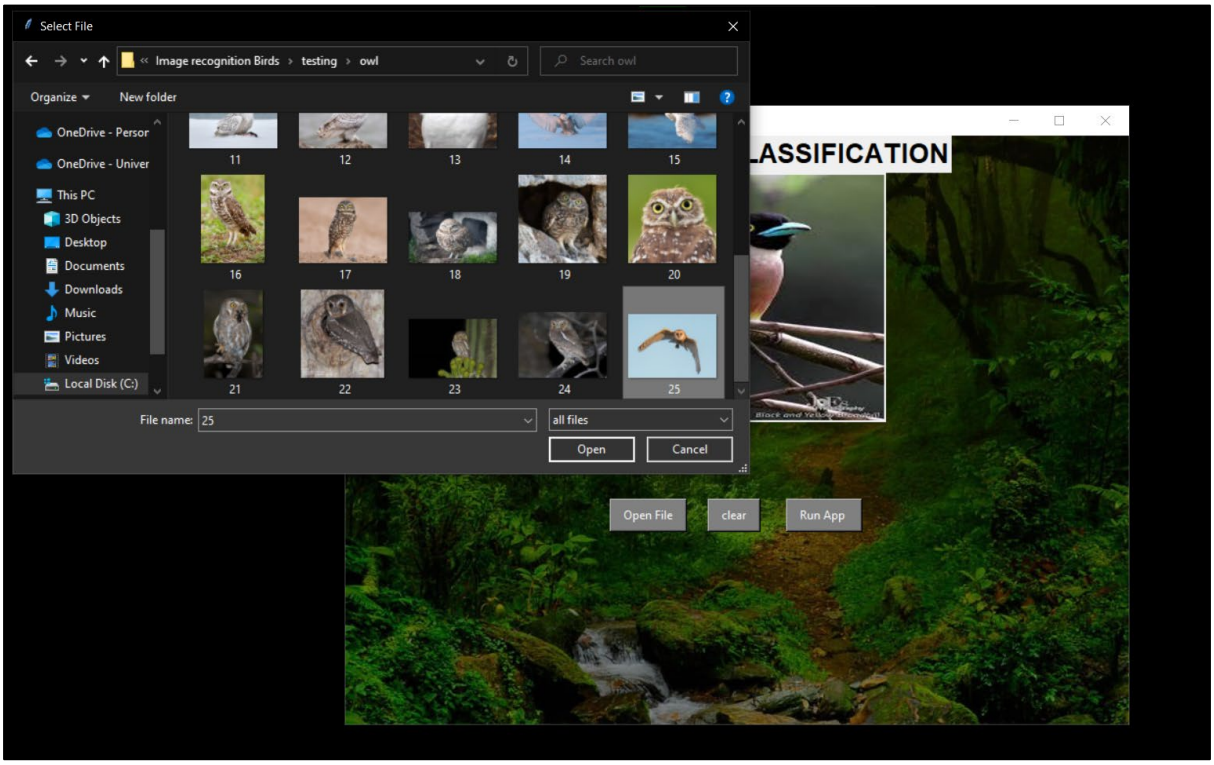

Figure 3.8.1 shows the input of choosing an image.

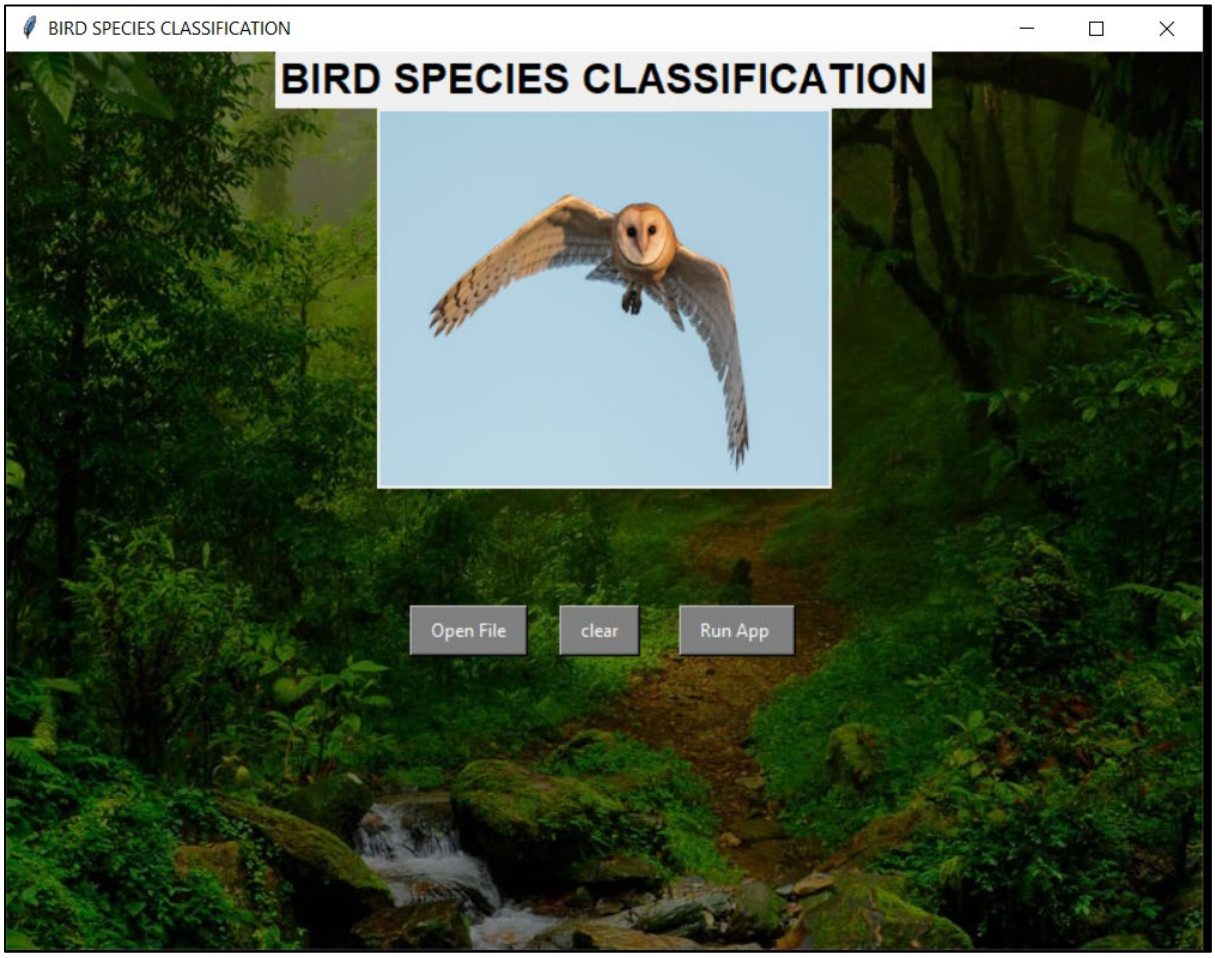

Figure 3.8.2 shows the input of uploading an image

3.2.2 OUTPUT FOR USER INTERFACE

According to Figure 3.8.3, this is the output generated after the user uploads the required image and clicks the 'Run App' button. The programme will present the available answer for the image selected. However, our system has a minor inaccuracy. This occurs when the bird species contains comparable qualities that the algorithm cannot recognise effectively, resulting in misleading output. Because both bird species have same attributes, the algorithm occasionally misidentified owl and eagle. Furthermore, our testing accuracy is lower than the train accuracy. For example, this programme misread the Owl as Eagle based on Figure 3.8.3.

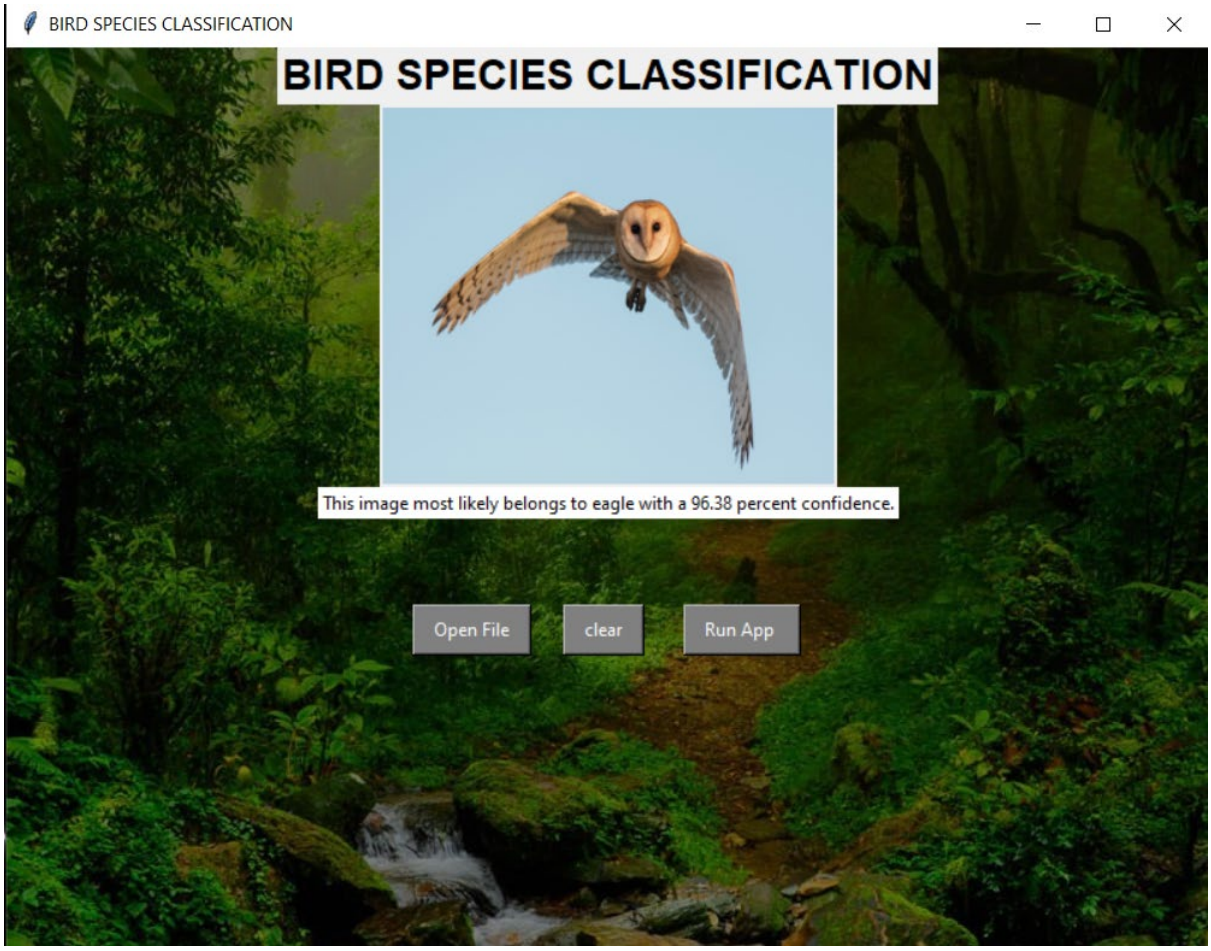

Figure 3.8.3 shows the output of the image

4.0 EXPERIMENTAL RESULTS AND ANALYSIS

In Bird species identification we utilize two Python files. One of these files is dedicated to deep learning, while the other's specifically used for training purposes. The next Python files that needs to be executed is the Graphical User Interface (GUI) responsible for handling input and displaying output results. Duirng the project, we conducted data training and testing as part of our work.

The range of datasets refers to the collection of different datasets. When the deep learning and training software is activated, it will show the training process from beginning to end. According to the data, this project has achieved an accuracy rate of over 75%. Due to its high accuracy, the project is capable of recognizing the majority of our images by their classifications. In addition, it should be noted that this algorithm is unable to identify images that do not belong to the classes included in the dataset.

The system is assessed during the experiment in order to identify the type of bird using the datasets we have collected. I find it much more enjoyable to figure out the result from the machine itself. During the trial, we had the opportunity to perform data training and testing using different datasets.
The datasets that can be used for data training and testing are presented in Tables 4.1 and 4.2. The experiment was conducted twice, resulting in an increase in the number of data sets provided for each trial. This was done to measure the training and testing accuracy outcomes.

Number of Training Datasets (Images)						
Experiment	broadbill	eagle	parrot	woodpecker	owl	Total
1	56	56	56	56	56	280
2	112	112	112	112	112	560

Table 4.1 shows the value of datasets for data training.

Number of testing Datasets (Images)						
Experiment	broadbill	eagle	parrot	woodpecker	owl	Total
1	14	14	14	14	14	70
2	28	28	28	28	28	140

Table 4.2 shows the value of datasets for data testing.

The first experiment's results show the dataset for data training, which consisted of 280 datasets, and, the second experiment's results, which involved a total of 560 datasets, are displayed in Table 4.1.

The first experiment's results show the dataset for data testing, which consisted of 70 datasets, and, the second experiment's results, which involved a total of 140 datasets, are displayed in Table 4.2.

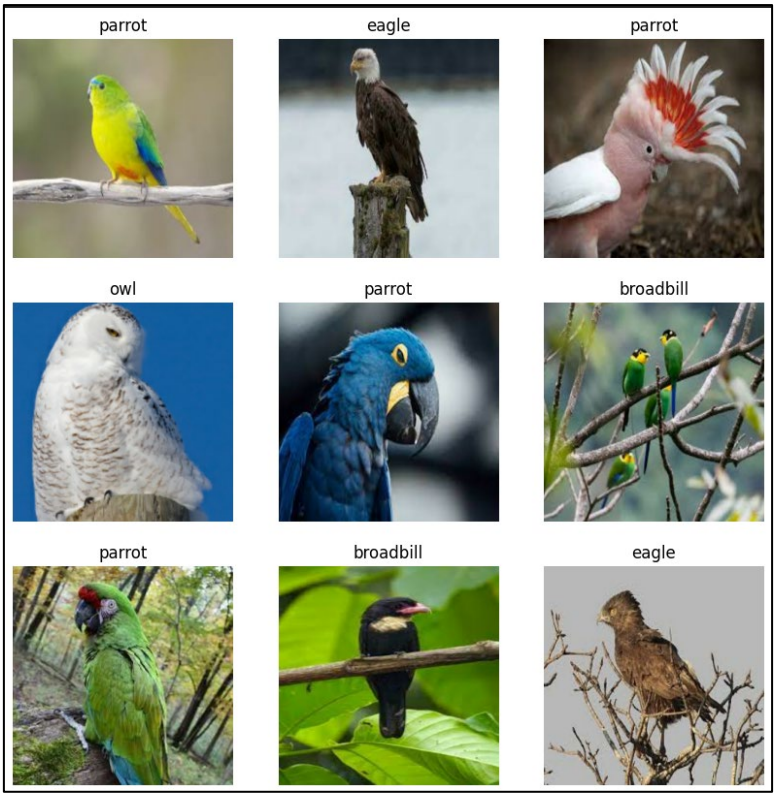

Figure 4.1 shows the output for the first experiment.

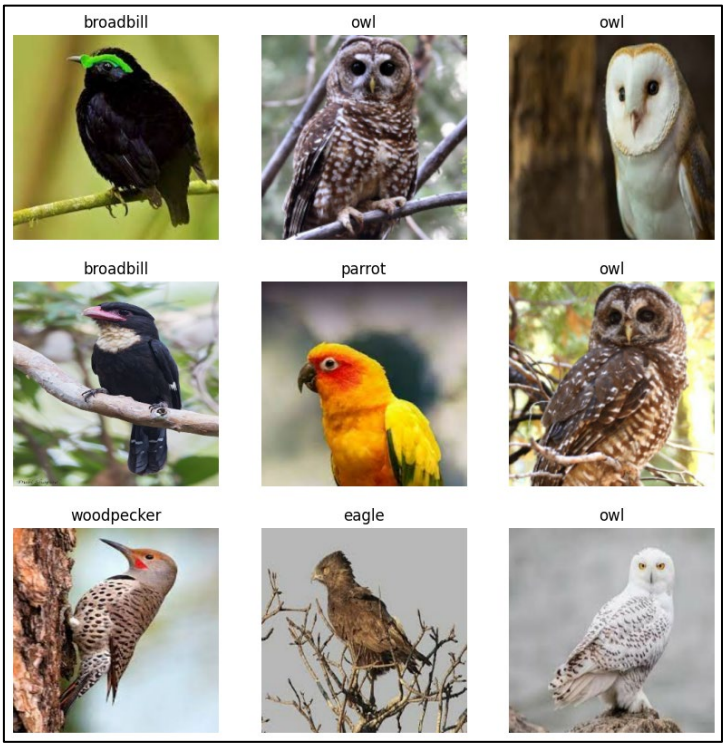

Figure 4.2 shows the output for the second experiment.

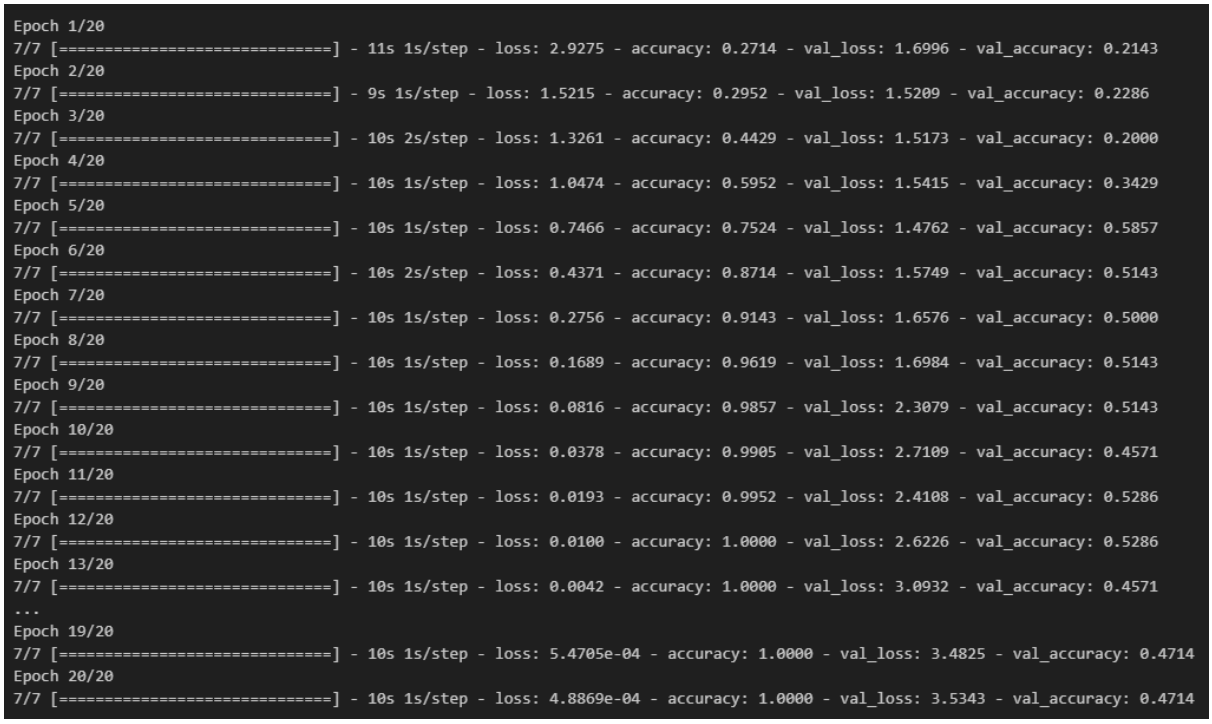

Figure 4.3 shows the epoch of the first experiment.

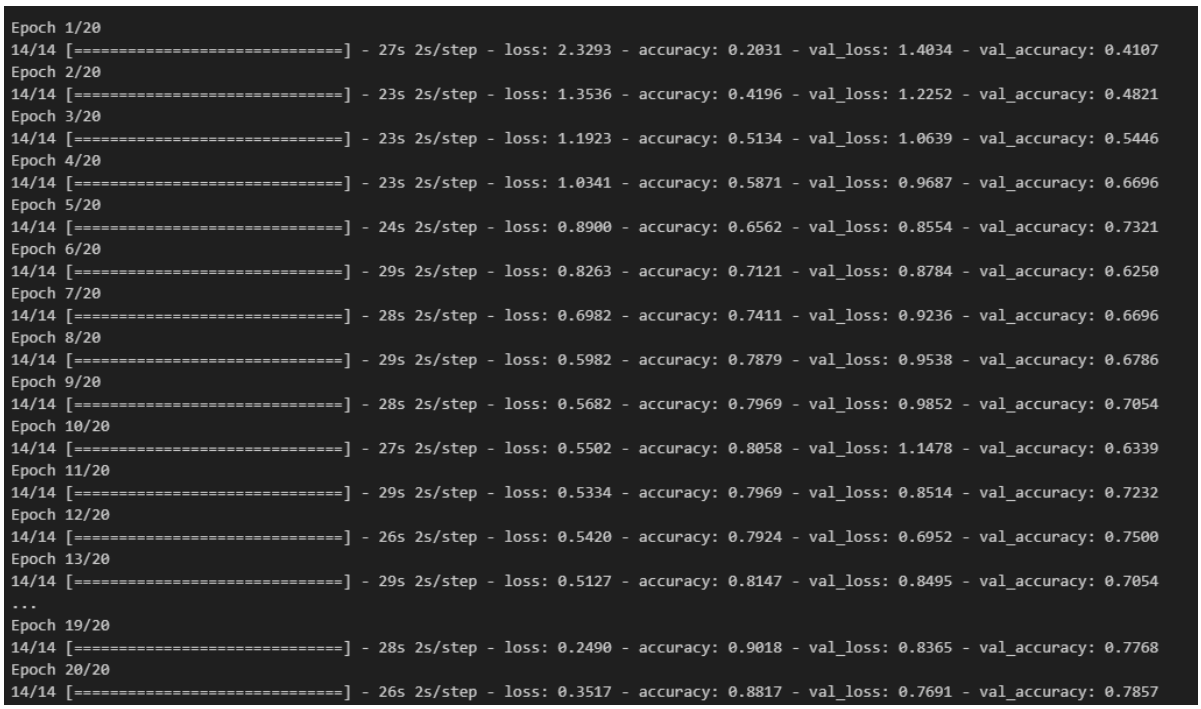

Figure 4.4 shows the epoch of the second experiment.

Every experiment consists of 20 epochs, and each epoch has its own distinct accuracy. The program has a very high accuracy, which means it can recognize most of our images based on their classes. Nevertheless, our program still has some shortcomings. After executing the program and conducting multiple tests using various images, it was observed that not all of the images were able to be recognized. The outcome produced by the program is somewhat imprecise. Based on our program's experience, the system is able to identify the category of bird species by analyzing the attributes of the bird species in the image.

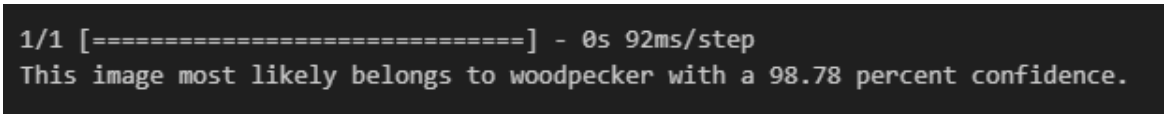

Figure 4.5 shows the accuracy percentage of the first experiment.

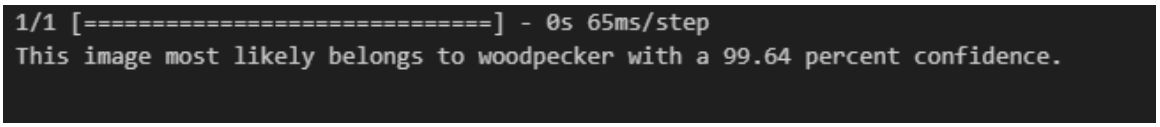

Figure 4.6 shows the accuracy of the second experiment.

After training the model with the datasets, the outcome of the two experiments is remarkable. The model has the ability to make predictions about the type or class of an object by analyzing an image. Based on the results of all the tests conducted, our model has shown the highest accuracy in identifying the type of apparel based on the image among all the matched images. However, it was observed that there was a difference in the accuracy of each experiment between the two.

In addition, the differences in the accuracy of the number of experiments can be attributed to changes in the learning algorithm. The reason is not because of a problem, but rather because of the features. Certain algorithms can be classified as unpredictable instead of deterministic. The outcome is that the unpredictable machine learning algorithm acquires a slightly distinct model with each execution on the identical data. The model might produce predictions that are somewhat different and exhibit slight variations in terms of error or accuracy. Despite the fact that the outcomes of each experiment differ, they all demonstrate similar patterns of growth. In order to determine the certainty values, we made the decision to gather multiple accuracy readings. To determine the average for each experiment, we collected five readings. It is important to obtain the average performance of the models by fitting multiple final models to your dataset. This can be done by averaging their forecasts when making predictions on new data.

Additionally, it arises due to a distinct platform. In the initial experiment, we assess the precision of the training data on Interactive Python Notebook (ipynb) by using Visual Studio Code, for instance. During the experiment, it was observed that the accuracy of the training data exhibited variation.

The reason for this is that Interactive Python Notebook utilizes the latest version of Python. We decided to use that Interactive Python Notebook on Visual Studio Code for running our model, regardless of the Python version. This choice was made because it allows group members to easily review the code, help each other when errors arise, and share the same document for importing datasets.

5.0 CONCLUSION

This project talked about how CNN can use the Keras API and Tensorflow to classify pictures. Convolutional neural networks are a popular deep learning method used for visual recognition tasks today. CNN is the best artificial neural network. It is used to model images, but it can also be used for a lot of other things. CNN, like all deep learning systems, depends a lot on how much and how good the training data is. When given a well-prepared dataset, CNNs can do better at visual recognition jobs than people. However, they can still be affected by things like glare and noise that people can deal with.

The theory of CNN is still developing, and researchers are working towards enhancing it with features like active attention and online memory. These improvements will enable CNNs to analyse new items that are quite different from what they were trained on. The resemblance to the mammalian visual system in this research suggests that it could contribute to the development of a more intelligent artificial visual recognition system. The convolutional layer is an important component of the convolutional neural network, which is why the network is named after it. The purpose of this layer is to perform a "convolution" operation.

In the context of a convolutional neural network, a "convolution" refers to a linear process. This process involves multiplying a set of weights with the input, which is similar to what happens in a standard neural network. The approach was designed for two-dimensional input. In this method, the multiplication is performed between an array of input data and a two-dimensional array of weights, which is commonly referred to as a filter or a kernel. The size of the filter is smaller compared to the input data, and the multiplication method used is a dot product. This dot product is performed between a patch of the input data that is the same size as the filter, and the filter itself. The dot product is obtained by multiplying each element of the input with the corresponding element of the filter's patch, and then adding all these products together to obtain a single value. The process is called the "scalar product" because it gives a single value.

The reason for choosing a smaller filter than the input is on purpose because it enables us to use the same set of weights to multiply with the input array at various positions. The filter is applied in a systematic manner to every overlapping component, or an area the size of the filter, of the incoming data. This is done from left to right and top to bottom. Accuracy plays a crucial role in evaluating the effectiveness of image classification, even though it may not always result in complete images. In reality, when conducting research with an uneven sample of data, the likelihood of accuracy decreases.

6.0 FURTHER WORK RECOMMENDATION

Our Bird Species Classification application is based on a CNN model that can process images and identify the species of birds. However, the performance of the model can be further improved by adjusting some of its parameters. One of the main parameters that can affect the performance of the model is the size of the dataset that is used for training. Currently, our dataset is limited in terms of the number of images and the diversity of bird species. Therefore, for the future development of our application, we may consider using a larger dataset that contains more images and more classes of birds. This can help the model to learn more features and patterns, and to recognize more species accurately. A larger dataset can also reduce the risk of overfitting and improve the generalization ability of the model. By using a larger dataset, we hope to enhance the performance of our Bird Species Classification application and provide a better service to our users.

Another option that we can explore for improving our Bird Species Classification application is to experiment with the amount of epochs and learning rate for our CNN model. These are two important parameters that can affect the speed and accuracy of the model's convergence. However, we should be careful not to use too many epochs, as this could cause the model to overfit the training data. Overfitting means that the model memorises the data rather than learning it, and performs poorly on new data. To avoid overfitting, we can monitor the accuracy of validation data for each epoch or iteration and stop the training when it starts to decrease. We can also use other techniques to prevent overfitting, such as adding noise to different parts of the model, such as dropout or batch normalisation. These techniques can help the model to learn more robust features and patterns, and to generalise better to new data. We can also choose a suitable batch size that balances the stability and efficiency of the model. Our goal is to use a CNN model to classify bird species from images, and to provide a useful and reliable service to our users.

Our Bird Species Classification application is a tool that can identify the species of birds from images. The application uses a simple and user-friendly interface that allows users to upload images and get the results. However, the UI of our application can be improved to make it more visually appealing and engaging. Therefore, for the future development of our application, we may consider improving the UI of our application by using different colors, fonts, icons, and animations. We may also add more features and functionalities to the UI, such as displaying more information about the bird species, allowing users to share their results on social media, or providing feedback and suggestions to the users. By improving the UI of our application, we hope to attract more users and increase their satisfaction and interest in our application and in birds.

7.0 REFERENCES

1. blogadmin. (2023, May 15). - OVHcloud Blog. OVHcloud Blog. https://blog.ovhcloud.com/
2. Jinde Shubham. (2018, July 14). What exactly does CNN see? - Becoming Human: Artificial Intelligence Magazine. Medium; Becoming Human: Artificial Intelligence Magazine. https://becominghuman.ai/what-exactly-does-cnn-see-4d436d8e6e52
3. Introduction to Convolutional Neural Networks | Rubik's Code. (2018, February 26). Rubik's Code. https://rubikscode.net/2018/02/26/introduction-to-convolutional-neural-networks/
4. Saha, S. (2018, December 15). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. Medium; Towards Data Science. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
5. Bold, N., Zhang, C., & Akashi, T. (2019). Bird Species Classification with Audio-Visual Data using CNN and Multiple Kernel Learning. https://doi.org/10.1109/cw.2019.00022
6. Kumar, M., Arun Kumar Yadav, Kumar, M., & Yadav, D. (2023). Bird Species Classification from Images Using Deep Learning. 388–401. https://doi.org/10.1007/978-3-031-31417-9_30
7. Ablikim, U., Ngoi, J., & Liu, P. (n.d.). Using CNNs to Recognize Bird Species by Song. Retrieved July 18, 2023, from http://cs230.stanford.edu/projects_winter_2021/reports/70723347.pdf
8. Sofya Lipnitskaya. (2021, January 3). Bird by Bird using Deep Learning - Towards Data Science. Medium; Towards Data Science. https://towardsdatascience.com/bird-by-bird-using-deep-learning-4c0fa81365d7
9. Manna, A., Upasani, N., Jadhav, S., Mane, R., Chaudhari, R., & Chatre, V. (2023). Bird Image Classification using Convolutional Neural Network Transfer Learning Architectures. International Journal of Advanced Computer Science and Applications, 14(3). https://doi.org/10.14569/ijacsa.2023.0140397
10. Dharaniya R, Preetha M, & Yashmi S. (2022, November 3). Bird Species Identification Using Convolutional Neural Network. ResearchGate; unknown. https://www.researchgate.net/publication/365080475_Bird_Species_Identification_Using_Convolutional_Neural_Network
11. mitra, krati. (2020, July). Bird Classification using CNN in PyTorch - Bird Classification using CNN in PyTorch - Medium. Medium; Bird Classification using CNN in PyTorch. https://medium.com/bird-classification-using-cnn-in-pytorch/bird-classification-using-cnn-in-pytorch-7329c172a508