

GitHub repo: <https://github.com/ucfnnbx/bird-classification>

## Introduction

In this project, transfer learning based on a pre-trained CNN model was used to build a classifier for bird species that are frequently spotted around the area of UCL East campus. Four types of birds were chosen to be classified in the current stage, which are crow, double breasted cormorant, Eurasian magpie and gull, as shown in Figure 1.

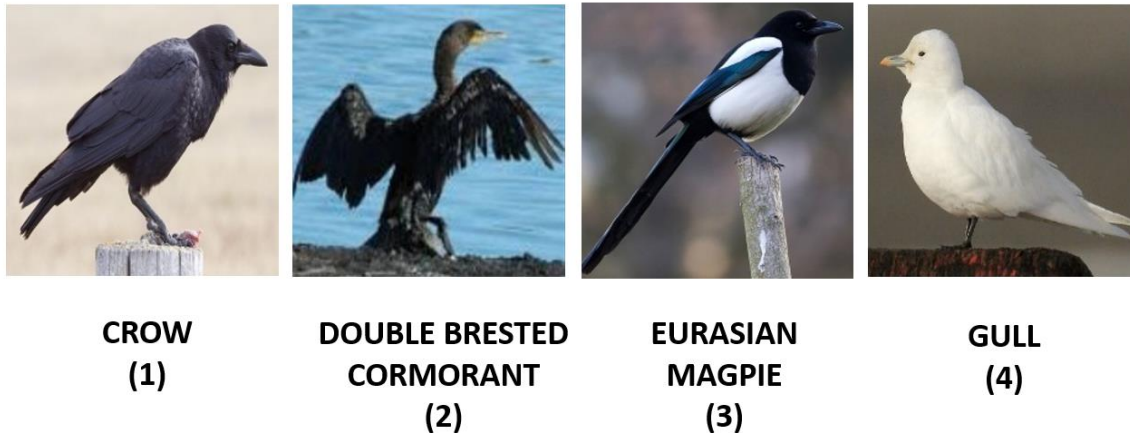


Figure 1. Four types of bird to be classified

The choice was made based on data from Aude's project, which shows the statistics of the 10 most detected bird species around the same area as this project.

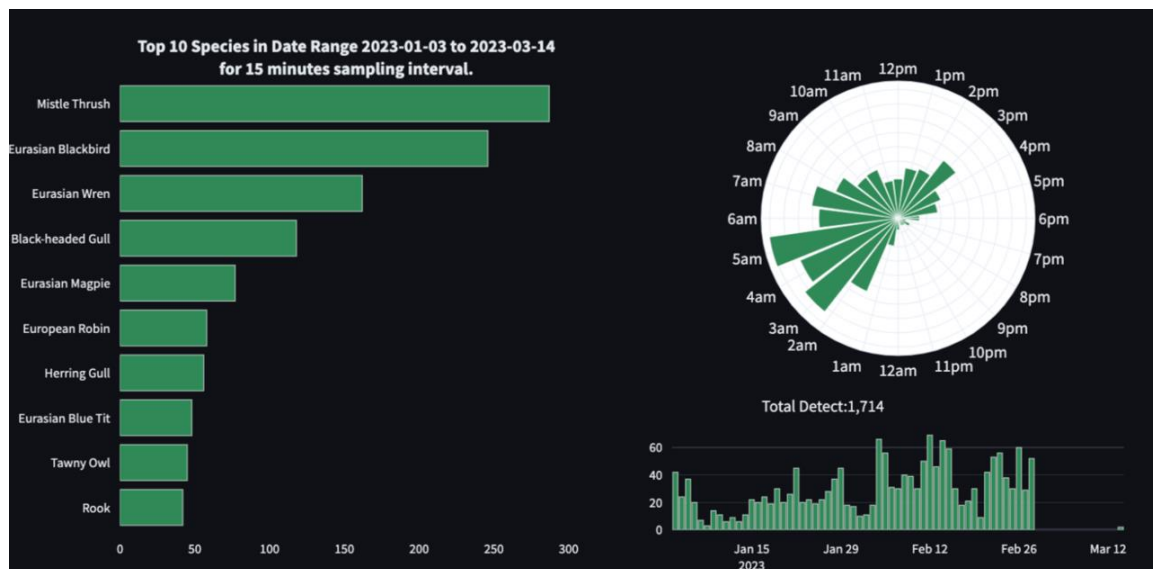


Figure 2. Statistics from Aude's project

After doing some literature review, there are currently three types of classifiers used for bird recognition: traditional machine learning, convolutional neural networks and transfer learning based on convolutional neural networks. It turns out that transfer learning usually returns a higher accuracy when the training dataset is relatively small (Alswaitti et al., 2022). Sharma et al. (2022) researched using transfer learning to identify bird species with both audio and video processing. The trained model was able to classify 137 kinds of bird with an overall accuracy of 90%. These prove that transfer learning is an optimal method used for bird recognition, especially when the dataset size is limited.

The target users for my project would be people who are interested in observing birds but have trouble recognizing types of bird on their own. With the model deployed on their mobile phones, whenever the users see a bird, they can know the name of bird just by taking a picture.

## Research Question

Can a CNN-based transfer learning model be used to classify main bird species around UCL East with good accuracy?

## Application overview

The platform used to train the model was Edge Impulse, which can store, tag and split the data; build, train and test the model; deploy the model to a mobile phone.

## Data

Describe what data sources you have used and any cleaning, wrangling or organising you have done. Including some examples of the data helps others understand what you have been working with.

The dataset consisting of images of four types of birds was chosen from Kaggle BIRDS 510 SPECIES dataset. Kaggle dataset has more than 150 images for each kind of bird. As shown below, birds are in different gestures and background therefore it is considered a diverse dataset. To add more diversity, I also added some self-shot images into the crow dataset, examples of which are in Figure 4.



Figure 3. Bird images in Kaggle dataset



Figure 4. Self-shot image added to 'crow' training data

Besides four types of bird images, I also added a fifth type of 'unknown' data consisting of random images that don't include birds. This is to avoid giving a bird type result when camera is shooting at a person, for example.

The training and testing data is split by 80/20 and validation data is 20 percent from training data.

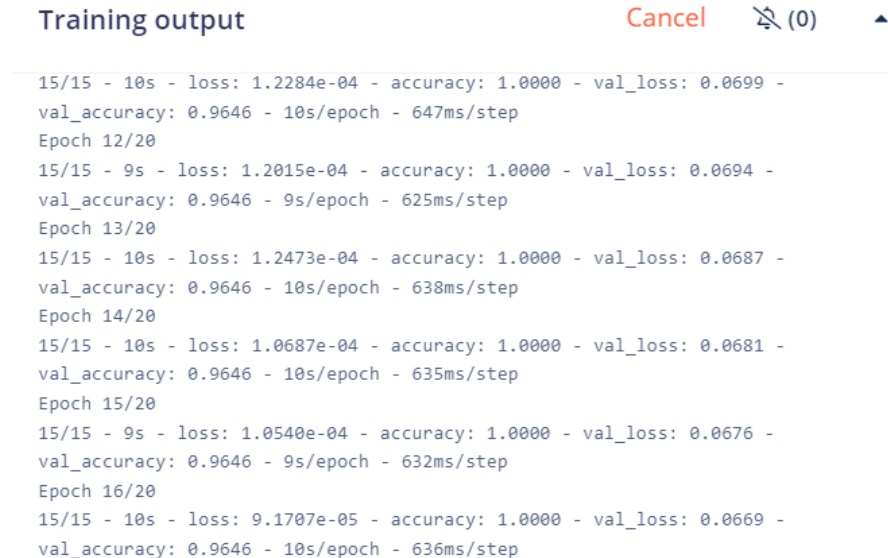
## Model

MobileNet-v2 160x160 0.35 model, a convolutional neural network that has 53 layers, was used in this project. The input size was chosen to be 160 by 160 instead of 96 by 96 because a higher input resolution results in a great improvement in accuracy. The width

multiplier is 0.35. A higher value is not recommended because the memory taken up would increase greatly but almost has no impact on the accuracy.

## Experiments

During the first training, the problem found was that the validation loss is nearly 100 times bigger than the training loss, as shown in Figure 5. This means that the model was overfitting. The reason was considered to be the model is too complex that it memorized the pattern of training data too much so that the model is unable to recognize new data and the accuracy would decrease. To solve this problem, I reduced the number of layers of the pretrained model and added dropout. Dropout is a technique to randomly neglect neurons during training which is an effective way to prevent overfitting (Brownlee, 2022).



```
Training output Cancel (0) ▲
15/15 - 10s - loss: 1.2284e-04 - accuracy: 1.0000 - val_loss: 0.0699 -
val_accuracy: 0.9646 - 10s/epoch - 647ms/step
Epoch 12/20
15/15 - 9s - loss: 1.2015e-04 - accuracy: 1.0000 - val_loss: 0.0694 -
val_accuracy: 0.9646 - 9s/epoch - 625ms/step
Epoch 13/20
15/15 - 10s - loss: 1.2473e-04 - accuracy: 1.0000 - val_loss: 0.0687 -
val_accuracy: 0.9646 - 10s/epoch - 638ms/step
Epoch 14/20
15/15 - 10s - loss: 1.0687e-04 - accuracy: 1.0000 - val_loss: 0.0681 -
val_accuracy: 0.9646 - 10s/epoch - 635ms/step
Epoch 15/20
15/15 - 9s - loss: 1.0540e-04 - accuracy: 1.0000 - val_loss: 0.0676 -
val_accuracy: 0.9646 - 9s/epoch - 632ms/step
Epoch 16/20
15/15 - 10s - loss: 9.1707e-05 - accuracy: 1.0000 - val_loss: 0.0669 -
val_accuracy: 0.9646 - 10s/epoch - 636ms/step
```

Figure 5. Overfitting because of too many layers in model

In the following training process, it was observed that after starting training, the training loss and validation loss start to decrease gradually. Then, starting from a certain epoch, the validation loss increases suddenly while the training loss continues decreasing. That's the sign when an overfitting starts. After observing the training output, I set the number of training epochs to be exactly when the validation loss and training loss start to converge. For example, in Figure 6, overfitting happens from epoch 14, so the training epoch is set to be 13.

Experiments were run to solve overfitting. Figure 6 shows the accuracy of three training, with disabling top 2 layers, 0.1 dropout (1) and disabling 3 layers, 0.2 dropout (2) and disabling 4 layers, 0.3 dropout (3). It was observed that only (3) did not have overfitting within 20 epochs. Also, in (1) and (2), the testing accuracy is lower than training accuracy, which means that overfitting happened. Therefore, (3) was used in the final model.

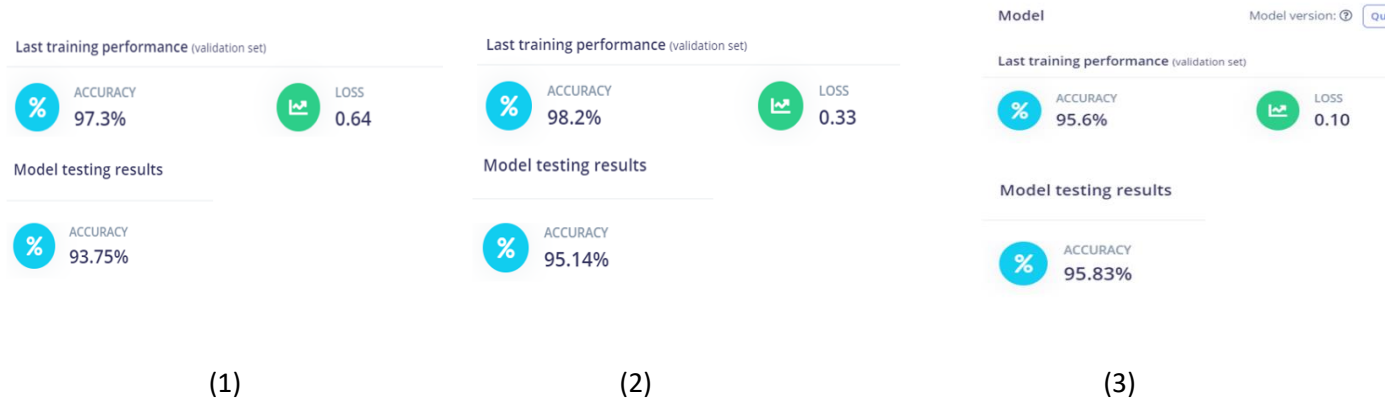


Figure 6. Accuracy of experiments to solve overfitting

## Data augmentation

Data augmentation is a process to randomly flip, crop, and change the brightness of images. Experiments on data augmentation show that it can allow to run more training cycles without overfitting. A comparison of training output is shown in Figure 7. The upper one without data augmentation reaches overfitting sooner than the lower one with data augmentation. The trained model's accuracy is shown in Figure 8, which also proves that overfitting leads to a decrease in accuracy, compared to the final model.

### Training output

```
Epoch 12/20
15/15 - 9s - loss: 0.0857 - accuracy: 0.9689 - val_loss: 0.1250 -
val_accuracy: 0.9381 - 9s/epoch - 603ms/step
Epoch 13/20
15/15 - 9s - loss: 0.0434 - accuracy: 0.9867 - val_loss: 0.1391 -
val_accuracy: 0.9558 - 9s/epoch - 577ms/step
Epoch 14/20
15/15 - 9s - loss: 0.0860 - accuracy: 0.9756 - val_loss: 0.1697 -
val_accuracy: 0.9381 - 9s/epoch - 577ms/step
Epoch 15/20
15/15 - 9s - loss: 0.0321 - accuracy: 0.9889 - val_loss: 0.1667 -
val_accuracy: 0.9469 - 9s/epoch - 604ms/step
Epoch 16/20
15/15 - 9s - loss: 0.0092 - accuracy: 1.0000 - val_loss: 0.1632 -
val_accuracy: 0.9558 - 9s/epoch - 578ms/step
```

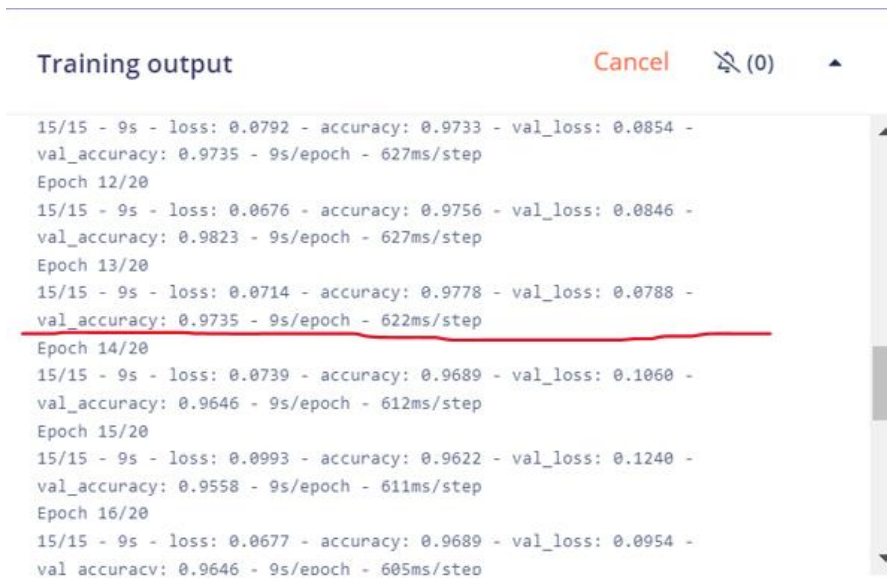


Figure 7. A comparison of training output with data augmentation

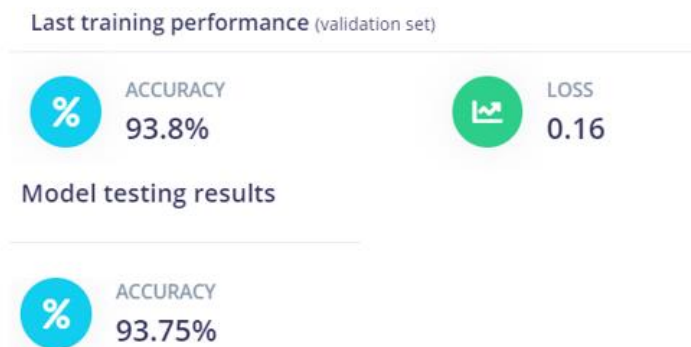


Figure 8. Accuracy without data augmentation

## Greyscale Preprocessing

Experiments were carried out on the impact of color of the image. The input images were pre-processed to be grey, and the training result is as below. The accuracy severely decreased because the color of birds is also an important feature to classify them.



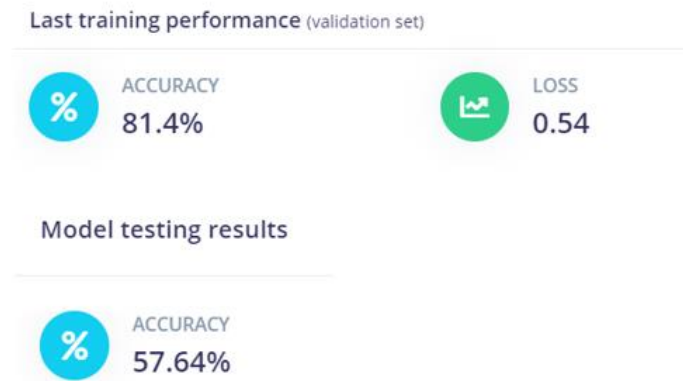


Figure 9. Greyscale image

## Learning rate

Through testing it was found that the higher learning rate, the less epochs needed for training loss and validation loss to converge. If the learning rate is too high, the accuracy decreases because it skips the optimal solution. 0.0005 is chosen to be the optimal learning rate here.

## Quantization

This is an experiment on quantization. It turns out that quantization can save 60% of flash usage with 1.7% less accuracy. It's worth doing this process if the model is going to be deployed on an Arduino Nano module.

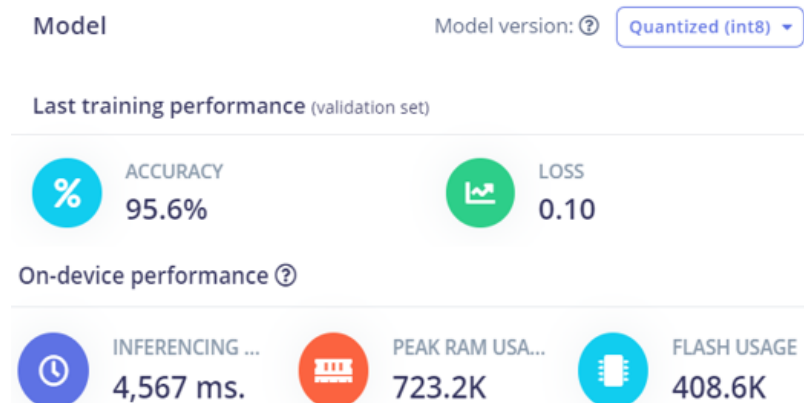


Figure 10. Performance after quantization

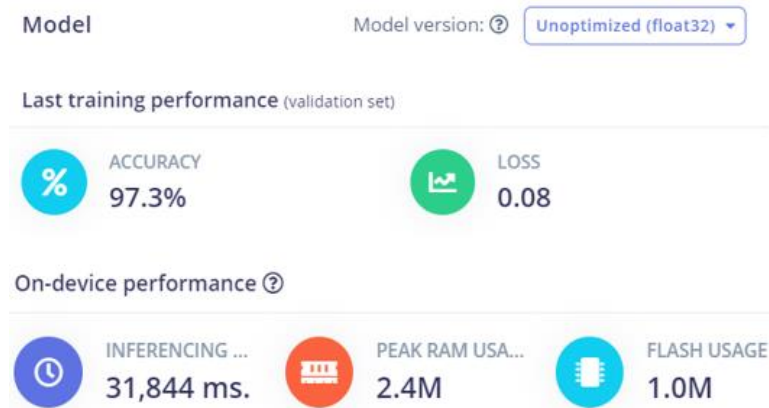


Figure 11. Performance before quantization

## Testing the model

The model was tested with a few self-shot images and a few images on the Internet by using mobile phones to take a picture of the image because I didn't have enough time to find real birds. The model turns out to work well in either case. Figure 12 and 13 are my self-shot images, two crows are in different gesture but the results are correct.

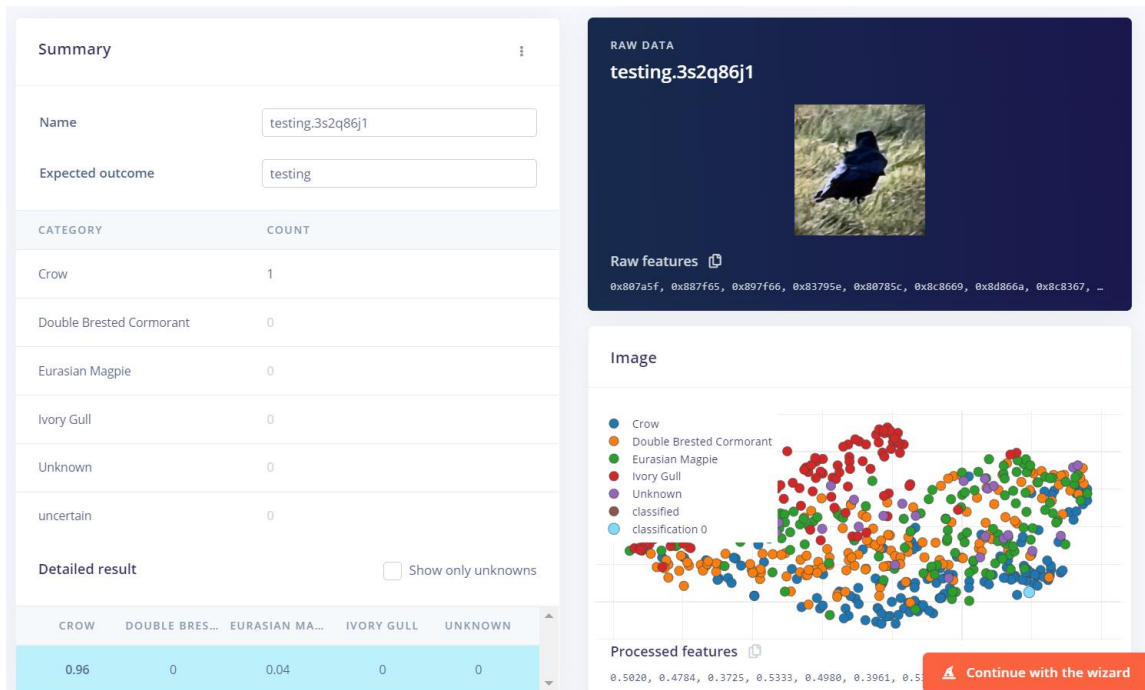


Figure 12. Testing result of a crow



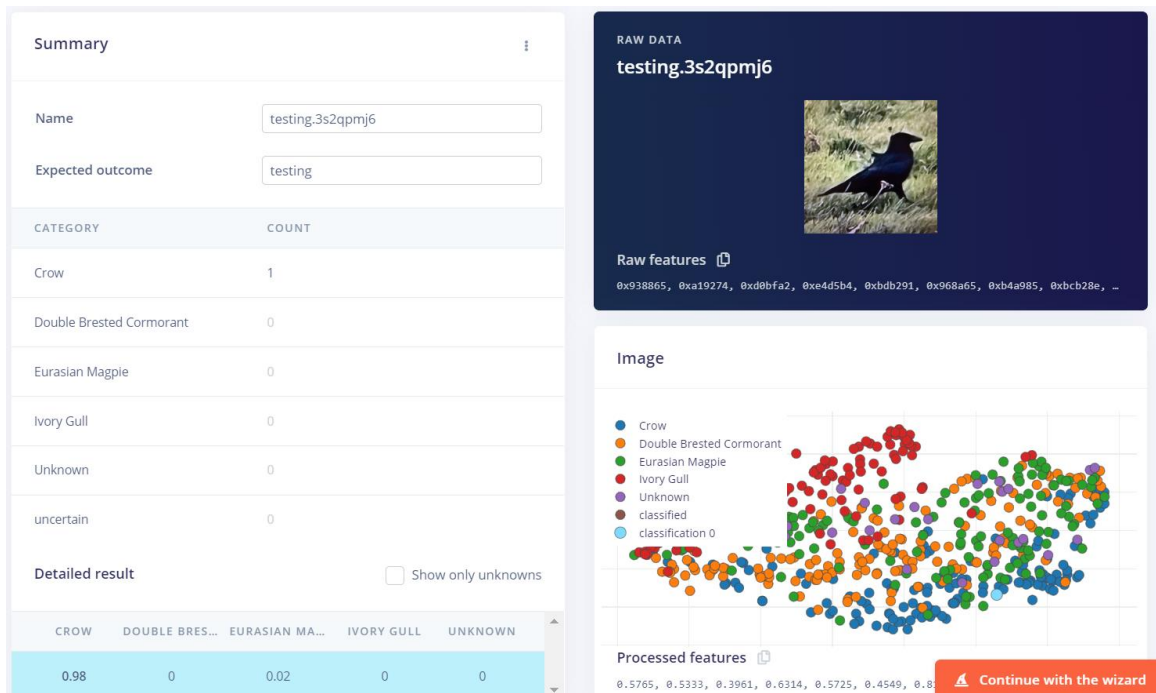


Figure 13. Testing result of a crow

Same in Figure 14 and 15. It is surprising to see that even though the image is blurred, the result is still accurate.

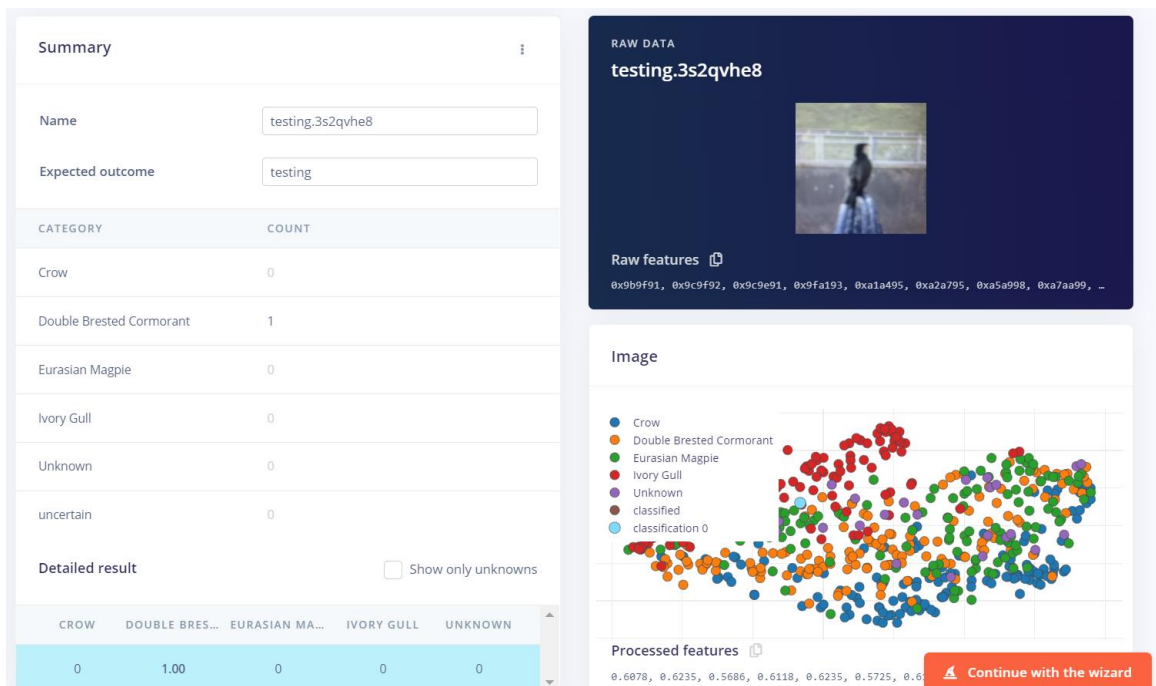


Figure 14. Testing result of a cormorant

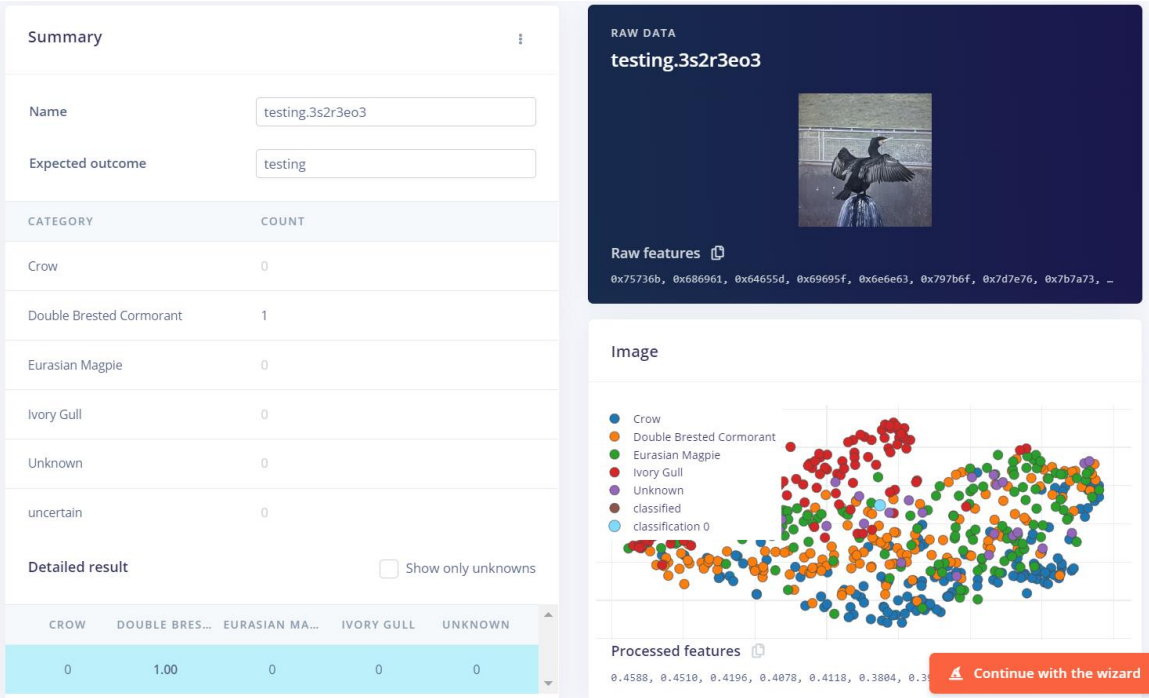


Figure 15. Testing result of a cormorant

This is testing of a self-shot gull.

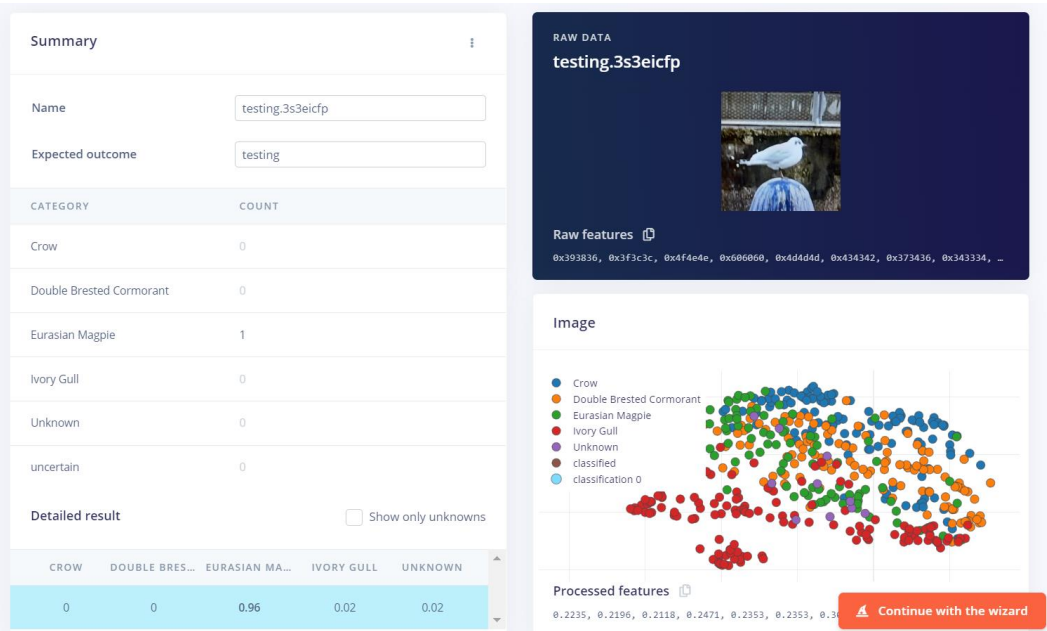


Figure 16. Testing result of a gull

The model also works well on classifying a random bird image on the Internet.

## Results and observations

### Last training performance (validation set)



### Model testing results



### Confusion matrix (validation set)

	CROW	DOUBLE BR	EURASIAN M	IVORY GULL	UNKNOWN
CROW	100%	0%	0%	0%	0%
DOUBLE BREST	3.3%	96.7%	0%	0%	0%
EURASIAN MAG	3.6%	3.6%	92.9%	0%	0%
IVORY GULL	0%	0%	0%	100%	0%
UNKNOWN	0%	0%	0%	0%	100%
F1 SCORE	0.95	0.97	0.96	1.00	1.00

### Feature explorer (full training set) ?

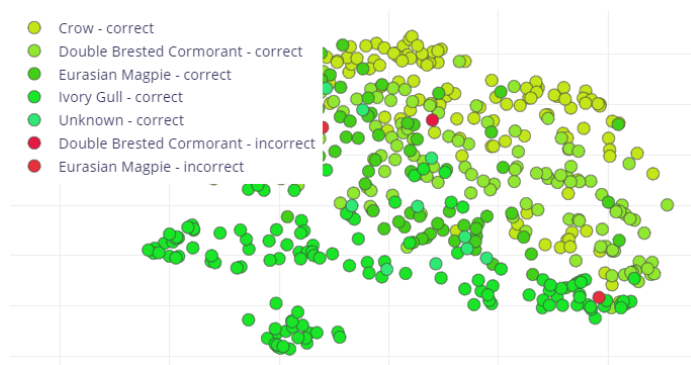


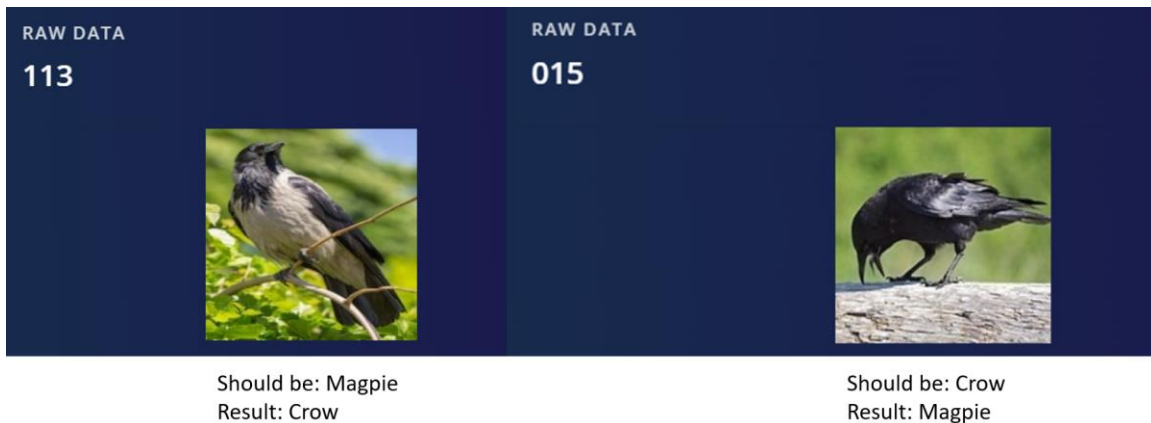
Figure 15. Confusion matrix and feature explorer

For the final model, it takes 10 minutes to train the model and 18 seconds to recognize an image, which might be because of the high resolution of input images and the complexity of model. Considering the high accuracy achieved, it is acceptable.

## Future steps

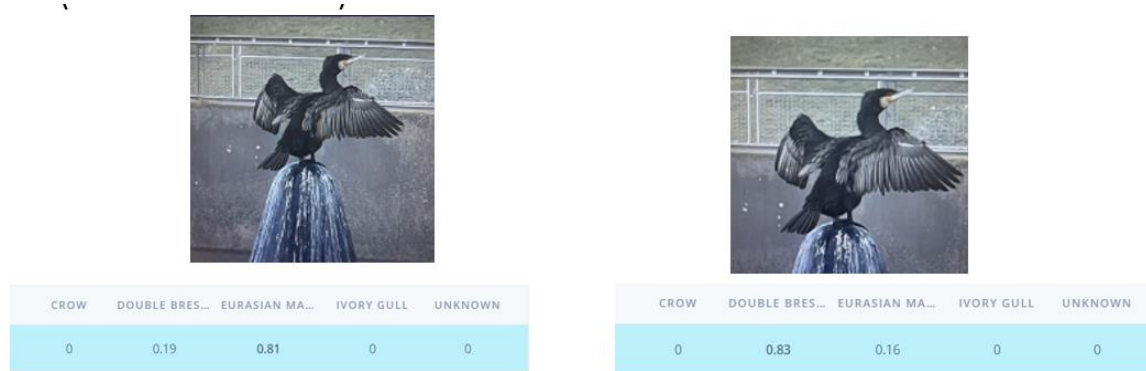
### Wrong testing results

Below is a wrong testing result example, in which the model fails to distinguish between magpie and crow. The reason might be they look similar especially when there is a reflection on the feather of the crow, it would hard to classify with color. To solve this, in the future more similar images could be added in the dataset to retrain the model.



### Limitation

One limitation found with the model is that the bird needs to take up most of the image and be centered or the result will be wrong. It might be because in the dataset all birds are in the center. To solve this, it will be possible to increase dataset with bird not in the center, which will achieve a more functional model.



## Reference

Alswaitti, M. *et al.* (2022) “Effective classification of birds’ species based on transfer learning,” *International Journal of Electrical and Computer Engineering (IJECE)*, 12(4), p. 4172. Available at: <https://doi.org/10.11591/ijece.v12i4.pp4172-4184>.

Brownlee, J. (2022) *Dropout regularization in deep learning models with Keras*, *MachineLearningMastery.com*. Available at: <https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/> (Accessed: May 4, 2023).

Sharma, N. *et al.* (2022) “Automatic identification of bird species using audio/video processing,” 2022 *International Conference for Advancement in Technology (ICONAT)* [Preprint]. Available at: <https://doi.org/10.1109/iconat53423.2022.9725906>.

## Declaration of Authorship

I, Yining An, confirm that the work presented in this assessment is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

*Yining An*

2023.5.4

