Project Title - Whales identification using Convolutional Neural Network

Problem Statement

For the purposes of observing whale populations, following their movements, and researching their behavior, individual whale identification is essential. However, it takes a lot of time and is prone to mistakes to manually identify humpback whales using their fluke patterns. Therefore, researchers and conservationists stand to gain significantly from the creation of an automated system that can precisely identify individual humpback whales.

Creating a CNN model that can precisely identify individual humpback whales from their fluke patterns would be the precise problem statement for this assignment. A big and varied dataset comprising a wide variety of photos of humpback whale flukes, each labelled with the matching whale ID, should be utilized to train the model. The whale should be recognised by the model with accuracy.

Dataset Description

Images of humpback whales taken by drones are included in the dataset for humpback whale identification. As humpback whales have distinctive markings on their tails called flukes, the dataset's goal is to help researchers and marine biologists identify specific whales.
A total of 25,000 photos of humpback whale flukes are included in the dataset, each of which has been labelled with the whale's unique individual identification number.
The dataset was preprocessed to eliminate unnecessary images, reshape them to a standard dimension, and augment it by horizontally and vertically rotating images.

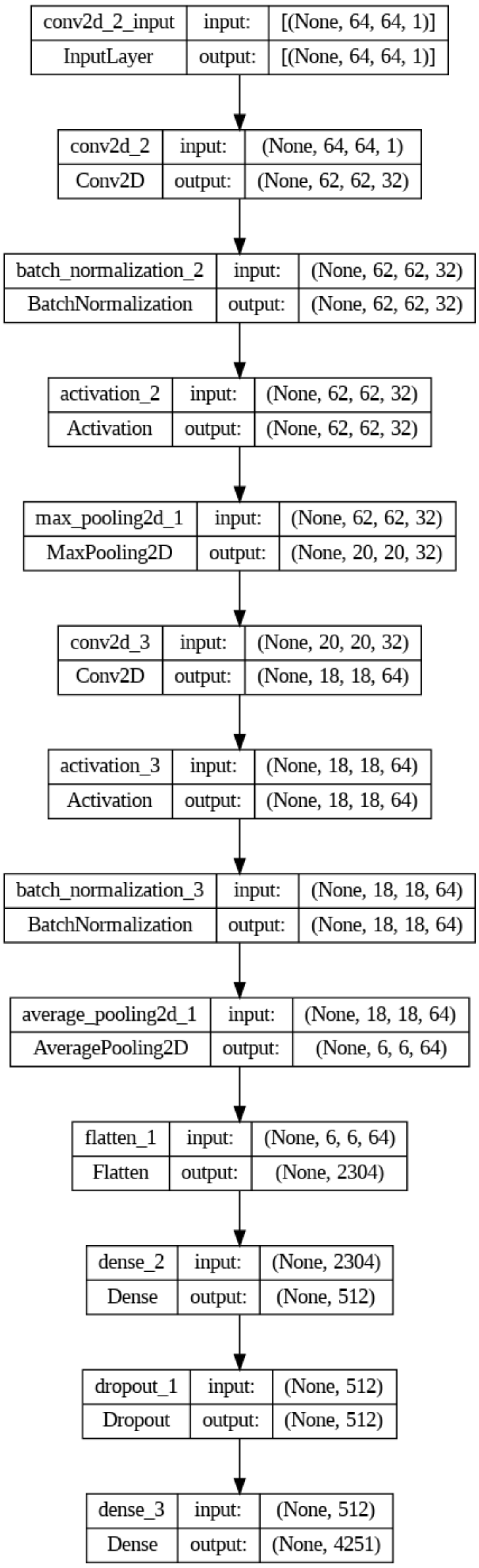
- The train set contains 9851 images.
- The test set contains 15,610 images.

Model Architecture

Convolutional neural networks (CNNs) could be used to identify certain whales based on their physical traits, such as the form and pattern of their flukes, as a potential solution to this issue. The deep learning method CNNs is a good choice for image identification tasks. To understand the distinctive characteristics of each whale’s fluke, they can be trained on a sizable dataset of whale fluke photos. Once trained, CNN can be used to automatically recognize whales in fresh images, speeding up, improving accuracy, and lowering intrusion. This technique can aid researchers in better understanding the dynamics of whale populations, tracking their whereabouts, and enhancing conservation efforts.

The **Architecture of our proposed Model** is:

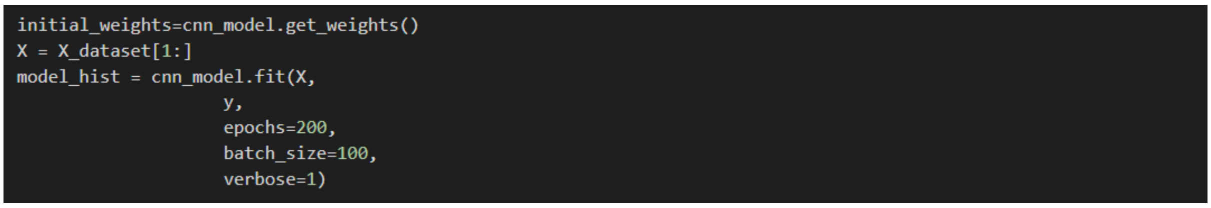
- *Input layer:* A 2D convolutional layer with 32 filters of size 3x3 and a stride of 1x1 serves as the input layer. The input shape option specifies the shape of an image in the dataset as the input shape.
- *Batch normalization layer:* The activations of the preceding layer in each batch are normalized.
- *ReLU activation layer:* The output of the preceding layer is subjected to the rectified linear unit activation function.
- *Max pooling layer:* Max pooling with a 3x3 pool size is performed on the output of the preceding layer.
- *Another 2D convolutional layer:* A 2D convolutional layer with 64 filters of size 3x3 and a stride of 1x1.
- *ReLU activation layer:* The output of the preceding layer is subjected to the rectified.
- *Batch normalization layer*: Normalizes the activations of the previous layer at each batch.
- *Average pooling layer:* Performs an average pooling operation with a 3x3 pool size on the output of the preceding layer.
- *Flatten layer:* Makes a 1D array out of the output from the preceding layer.
- *Fully connected layer:* 512 neurons are present in a dense layer, and the rectified linear unit activation function is used.
- *Dropout layer:* Reduce overfitting by adding a dropout layer that, at random, removes 50% of the outputs from the preceding layer.
- *Output layer:* To generate the final output probabilities for each class, a dense layer using the SoftMax activation function is used.


conv2d_2_input | InputLayer | input: [(None, 64, 64, 1)] | output: [(None, 64, 64, 1)]
conv2d_2 | Conv2D | input: (None, 64, 64, 1) | output: (None, 62, 62, 32)
batch_normalization_2 | BatchNormalization | input: (None, 62, 62, 32) | output: (None, 62, 62, 32)
activation_2 | Activation | input: (None, 62, 62, 32) | output: (None, 62, 62, 32)
max_pooling2d_1 | MaxPooling2D | input: (None, 62, 62, 32) | output: (None, 20, 20, 32)
conv2d_3 | Conv2D | input: (None, 20, 20, 32) | output: (None, 18, 18, 64)
activation_3 | Activation | input: (None, 18, 18, 64) | output: (None, 18, 18, 64)
batch_normalization_3 | BatchNormalization | input: (None, 18, 18, 64) | output: (None, 18, 18, 64)
average_pooling2d_1 | AveragePooling2D | input: (None, 18, 18, 64) | output: (None, 6, 6, 64)
flatten_1 | Flatten | input: (None, 6, 6, 64) | output: (None, 2304)
dense_2 | Dense | input: (None, 2304) | output: (None, 512)
dropout_1 | Dropout | input: (None, 512) | output: (None, 512)
dense_3 | Dense | input: (None, 512) | output: (None, 4251)

The model architecture, including the input and output shapes of each layer and the number of parameters in each layer, is displayed using the summary() function.

Model Training and Evaluation

Model Training: The CNN model is trained using the input data X and the target data y. The number of times the complete training dataset is run through the model during training is determined by the epoch's parameter. In this instance, 200 training epochs will be applied to the model.



```
initial_weights=cnn_model.get_weights()
X = X_dataset[1:]
model_hist = cnn_model.fit(X,
                    y,
                    epochs=200,
                    batch_size=100,
                    verbose=1)
```

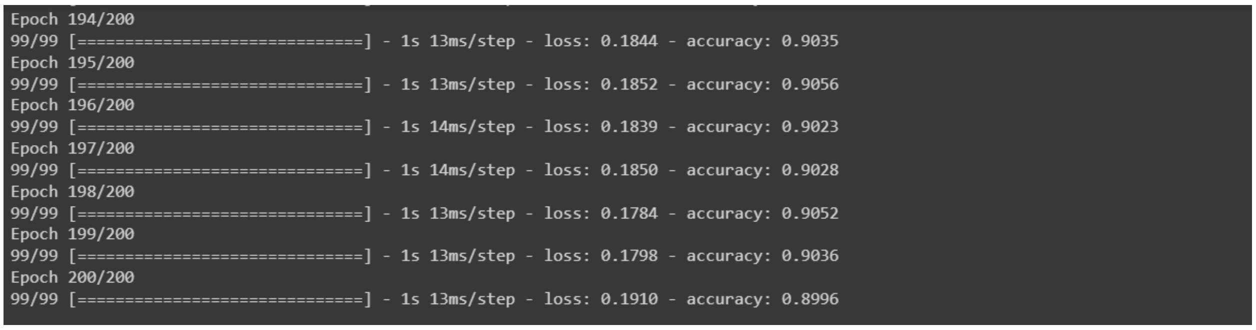
The batch size option determines how many samples are used in each iteration to update the model weights. Each iteration will consume 100 samples in this case. The verbose option regulates the training process's level of detail. A value of 1 indicates that throughout training, progress updates will be printed to the console.

The history object that fit() method returns contains details about the training procedure. The model's performance on the validation set is assessed and printed to the console at the conclusion of each epoch. Using the validation data parameter, the validation data is specified during model training.

As the trend for accuracy across each of the datasets keeps on increasing over the last few epochs, it can be seen from the accuracy plot that the model could likely be trained slightly further. After training is finished, the accuracy and loss during training and validation can be seen over the time.

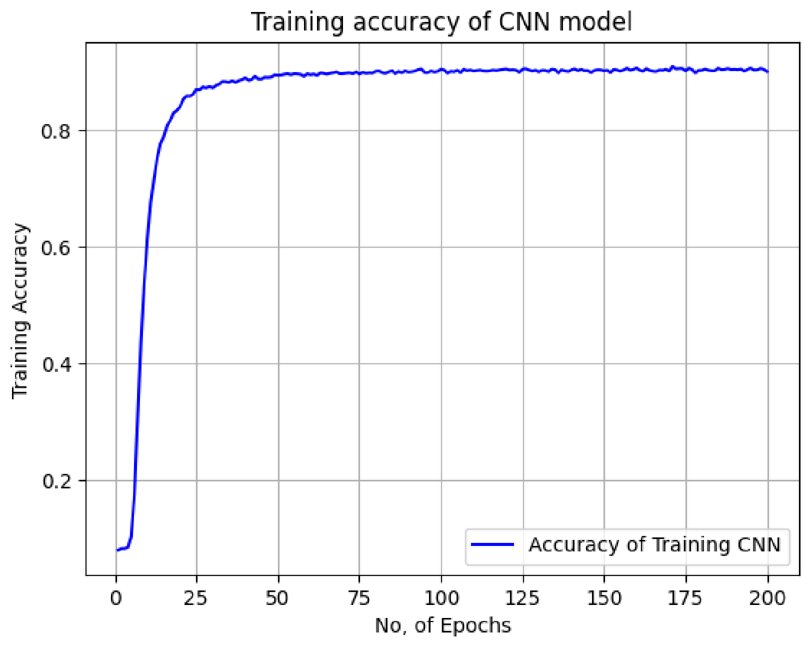
Training Results

Model Loss is 0.1910: This is the last epoch's loss function value. The model is doing better as the loss value decreases.



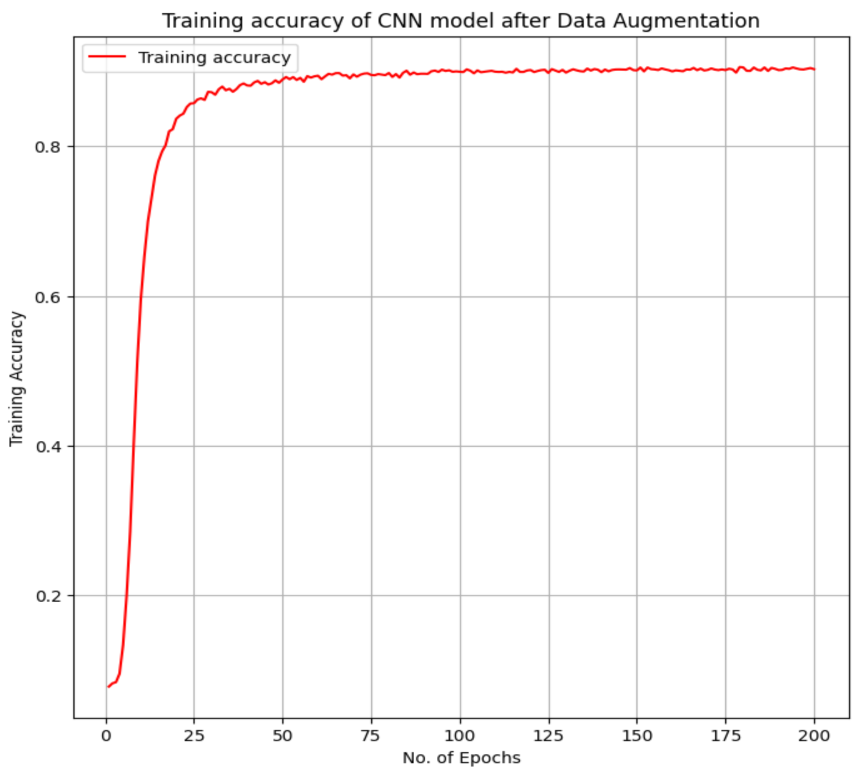
```
Epoch 194/200
99/99 [==============================] - 1s 13ms/step - loss: 0.1844 - accuracy: 0.9035
Epoch 195/200
99/99 [==============================] - 1s 13ms/step - loss: 0.1852 - accuracy: 0.9056
Epoch 196/200
99/99 [==============================] - 1s 14ms/step - loss: 0.1839 - accuracy: 0.9023
Epoch 197/200
99/99 [==============================] - 1s 14ms/step - loss: 0.1850 - accuracy: 0.9028
Epoch 198/200
99/99 [==============================] - 1s 13ms/step - loss: 0.1784 - accuracy: 0.9052
Epoch 199/200
99/99 [==============================] - 1s 13ms/step - loss: 0.1798 - accuracy: 0.9036
Epoch 200/200
99/99 [==============================] - 1s 13ms/step - loss: 0.1910 - accuracy: 0.8996
```

Model Accuracy is 0.8996% as clearly depicts in the below plot. Based on training data, this is the model's accuracy data; for the last epoch. The percentage of correctly categorized samples among all samples is the accuracy. The accuracy in this instance is 0.8996, or roughly 89.96%.



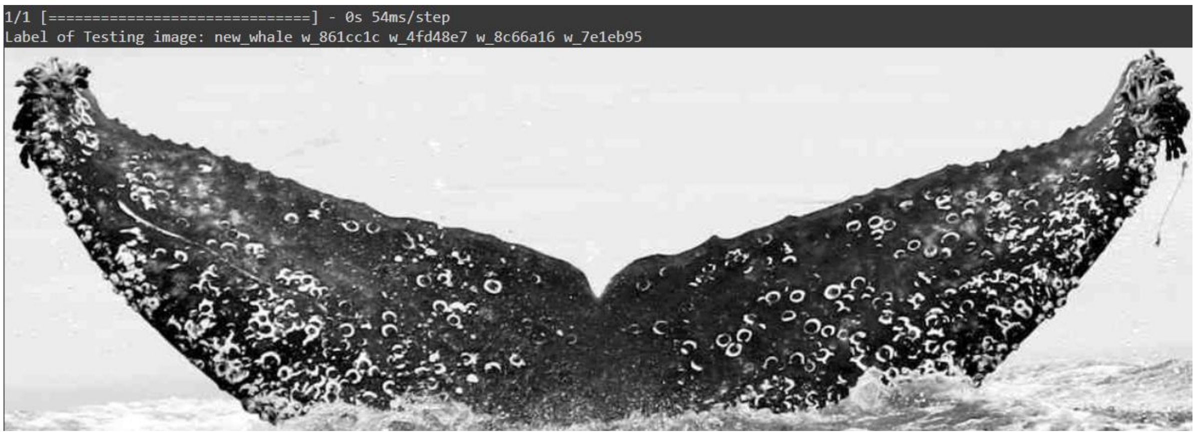
Training Accuracy using Data Augmentation

The data sets are generated through data augmentation are preferable as they can enhance the predictive accuracy and overall performance of machine learning models while lowering the risk of overfitting, which occurs when a model incorrectly identifies imprecise values in a dataset. This is a very effective approach when you have a very limited size of the dataset.



Testing Results

A proposed model or algorithm was used to predict the testing image. Based on the label, it appears that the image contains at least one known individual whale with the labels “new whale”, “w_861cc1c”, “w_4fd48e7”, “w_8c66a16” and “w_7e1eb95”.



```
1/1 [==============================] - 0s 54ms/step
Label of Testing image: new_whale w_861cc1c w_4fd48e7 w_8c66a16 w_7e1eb95
```

Conclusion

The implementation of a CNN-based automated approach to find particular whales based on their fluke patterns can contribute substantially to conservation efforts. Expanding the amount of the dataset, raising the model's depth, or employing transfer learning from previously-trained models can further enhance the model. This technology can assist researchers in gaining a deeper understanding of the behavioural patterns and population structure of whales, monitoring their movements, and enhancing conservation efforts.

References:

https://ieeexplore.ieee.org/abstract/document/9295729

https://www.kaggle.com/code/martinpiotte/whale-recognition-model-with-score-0-78563/input