COMP3330: Machine Intelligence
Semester 1, 2021

Project B:
Deep Learning Project Individual

Mirak Bumnanpol C3320409

1 Introduction

In this assignment, we will explore various training models and apply transfer learning to obtain training and evaluation data. This report details how the first task, Skin Cancer MNIST: HAM10000 Data, uses ResNet50 and MobileNet to run and compare two different training models. The second task, Plastic Waste in the Environment, is to develop a computer vision system which can detect and classify waste that can be collected. In order to achieve this, this report documents again, the use of different training models to obtain the relevant data which will then be evaluated on a secret test set. This report will go into further depth regarding the outcomes of the training models and the results which were achieved.

2 Skin Cancer MNIST: HAM10000 Data

For this task, we train a neural network to classify images of pigmented skin lesions for automated diagnosis. This task required that we run and compare two different model architectures, using both a larger and a smaller network to train the Skin Cancer MNIST: HAM10000 dataset. Using pre-trained models of ResNet50 as the larger CNN, and MobileNet as the smaller CNN, we compared features and differences of the two model architectures when used on the dataset.

For both models, the results of the analysis of data were first displayed in order to have a clearer picture of the statistics of the dataset which we were working with. As shown in Figure 1 and Figure 2, the balance of the results is skewered with lesion Melanocytic nevi and the localisation of the back as the most common lesion and area of the body affected with skin cancer.

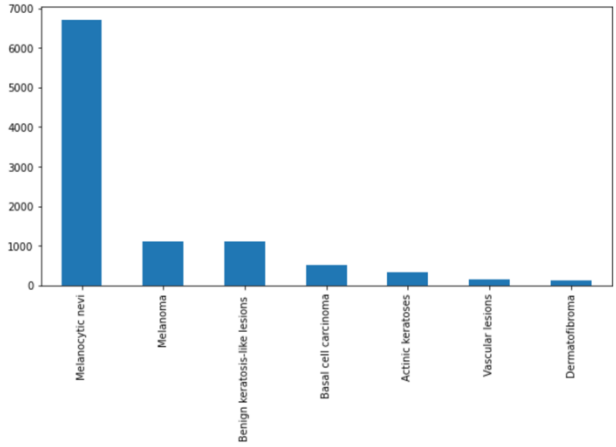

Figure 2: Skin Lesion Stats in Dataset

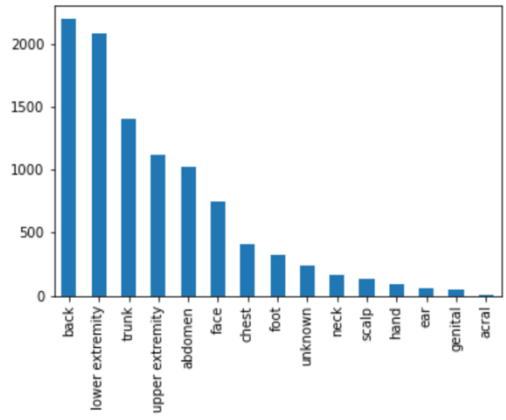

Figure 1: Localisation of Skin Lesions Stats

Both models used the same CNN as it was found to work more accurately. We also kept getting an error when trying to parse the dictionary for the key 'images', so a larger CNN with more classifications and batch normalisation was used instead of a smaller one. The CNN consists of a convolutional stack, which contains convolutional filters, batch normalisation, ReLu activations and max pooling. Batch normalisation is applied after the convolution but before activation. The model uses transfer learning to help support faster training time and is more efficient than training the models bottom up due to lack of computational hardware. The data is cross-validated with both a test set and a validation set to avoid the issue of overfitting. Overfitting is also avoided by utilising data augmentation (datagen function in the code) and the ImageDataGenerator method. The optimiser used was Adam and Keras was heavily utilised to build the CNN.

Both models were trained using fixed hyperparameters [Table 1] for consistency stake when comparing how each model performed against each other. We experimented a lot with max epoch sizes, batch sizes and the learn rate in order to test the models for the best results.

Fixed hyperparameters	Values
Dropout	0.5
Max Epochs	30
Batch Size	128
Patience	2
Learn Rate	0.001
Decay	0.0

Table 1: Hyper parameters used for training the models

2.1 Experiments and Results

The experiments yielded surprising results from the ResNet50 and the MobileNet networks. The ResNet50 pre-trained model is a larger network which we hypothesised would be a faster and more accurate than MobileNet. For ResNet50, the 30 epochs took an average of approximately 24.87 seconds to train. As a cross-validation method was used, we are able to see that the highest training accuracy of the model was 99.85% but once the model is evaluated, the average validation accuracy becomes 76.15% with loss at 1.1201. The ResNet50 model trained fast but needed a better validation accuracy. Below are Figures 3 and 4 which present the accuracy and loss graphs for the training model trained with ResNet50.

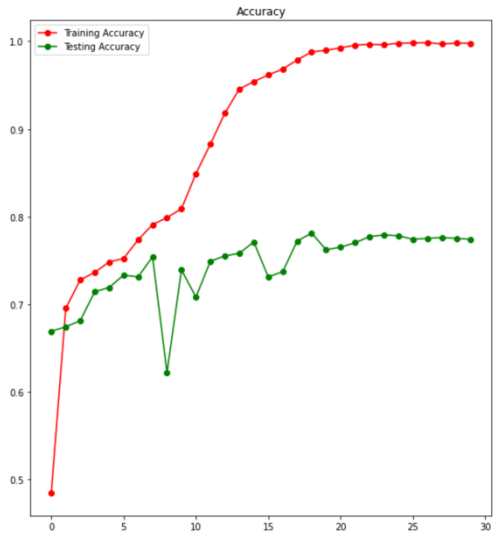

Figure 3: Accuracy Graph for ResNet50

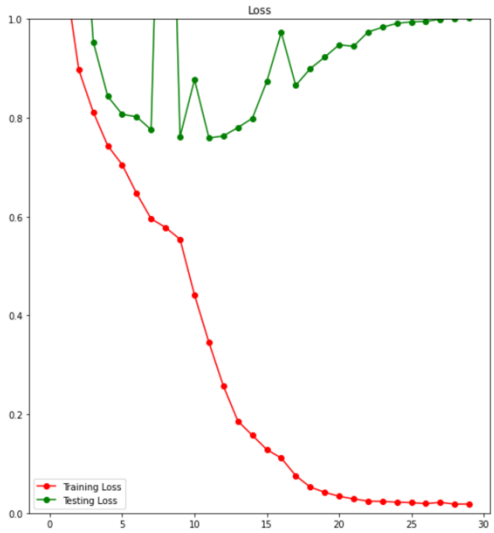

Figure 4: Loss Graph for ResNet50

As shown, it is clear that training accuracy shows an upward curve until stabilising towards the end at around 99.70% while simultaneously the training loss curve shows a downward curve and also stabilises towards the end. The testing accuracy and loss are both erratic, demonstrating why the validation accuracy is much lower than the training accuracy.

Meanwhile, the MobileNet pre-trained model uses a much smaller network as it is normally used for mobile applications. The 30 epochs took an average of approximately 29.77 seconds to train. Surprisingly although MobileNet is a much smaller network than ResNet50, the average validation accuracy for the MobileNet sits at 75.15% and loss at 1.0511. The highest training accuracy is at 99.35%. These results are surprising as MobileNet took 5 seconds longer to train 30 epochs than ResNet50 but the accuracy and loss are quite similar. Hence, we are able to see MobileNet still has a big enough capacity to learn and train the task.

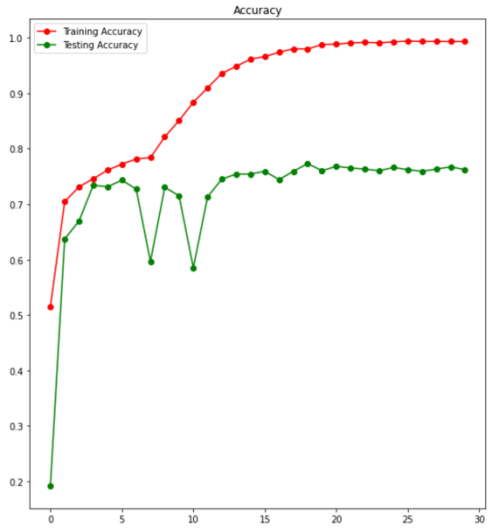

Figure 5: Accuracy Graph for MobileNet

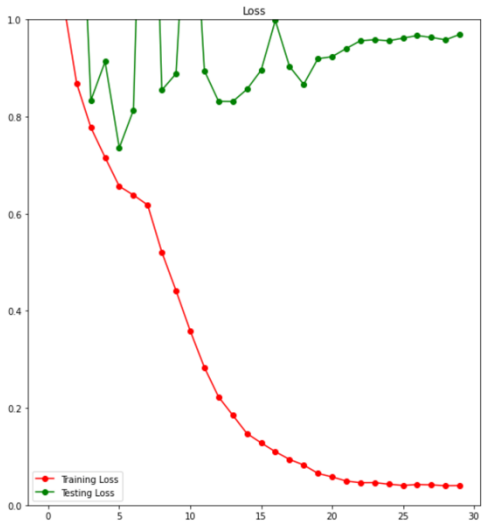

Figure 6: Loss Graph for MobileNet

The accuracy and loss graph for MobileNet is very similar to that of ResNet50 but shows more of a smoother curve. The testing accuracy and loss for MobileNet is more volatile than that of ResNet50 but the lines are closer to the training lines which results in a lower loss.

It was also noted during experiments that if both models trained for longer, the generalisation error only continues to grow due to the overfitting issue. The learning rate was also very important for both models as having a higher learning rate leads to prolonged training time but a smaller learning late results in an underperforming model.

Overall, it was clear to see that ResNet50 trained faster and more accurately than MobileNet.

3 Plastic Waste in the Environment

This task involved developing a computer vision system for a fleet of autonomous ground robots and air drones which are capable of detecting and classifying waste into four classes: plastic bags, plastic bottles, other plastic waste, and no plastic in the image. Based off the model's evaluation, the waste can then be collected.

3.1 Collecting Data

Although we were provided a selection of images on Blackboard to use for this task, to more effectively train our model we sought to find alternate ways to obtain data for the given task. We collected additional images of

plastic bottles from Google Image searches, using targeted keywords to find appropriate data. Alongside this, we took some of our own photos in a variety of different outdoor environments, including parks, the beach, and the University campus. Combining the data, we had 200 images total, with approximately half containing plastic waste. This more diverse dataset is better to train the models in order to further our chances with the secret test set.

3.2 Training the Network

For this task we chose to develop our own convolutional neural network model using Keras, which allowed us to build the model layer by layer using the Sequential class. In the figure below, the architecture of the CNN model is detailed.

Layer (type)	Output Shape	Parameters
Conv2D	(None, 198, 198, 16)	448
Activation (ReLU)	(None, 198, 198, 16)	0
MaxPooling2	(None, 99, 99, 16)	0
Conv2D	(None, 97, 97, 32)	4640
Activation (ReLU)	(None, 97, 97, 32)	0
MaxPooling2D	(None, 48, 48, 32)	0
Conv2D	(None, 46, 46, 64)	18496
Activation (ReLU)	(None, 46, 46, 64)	0
MaxPooling2D	(None, 23, 23, 64)	0
Flatten	(None, 33856)	0
Dense	(None, 512)	17334784
Activation (ReLU)	(None, 512)	0
Dense	(None, 4)	2052
Activation (Softmax)	(None, 4)	0

Table 2: Convolutional Neural Network

Due to a lack of available computational resources and time constraints, we minimised the total dataset count to the initial 40 images provided in Blackboard. Ideally, we wanted to train a broader set of data with transformations such as rotations, colour shifting, and image flipping so that the model could detect plastic waste objects with greater accuracy.

The model was compiled with a categorical cross-entropy loss function, since we are dealing with more than 2 label classes.

3.3 Experiments and Results

After splitting the dataset into testing, training, and validation data and building our model, we ran a variety of experiments both as a group and independently.

We experimented with different batch sizes, however, did not see a significant difference between the values used. The batch size which was eventually used was 6 as it was the best and most effective size to use when training this model. RMSprop was chosen as the final optimisation function for the project, with the default learning rate applied. Throughout experimentation, we also used the Adam optimisation function, however it performed worse than RMSprop. Hence, the most successful result was RMSprop. The below Figures 7 and 8 present the accuracy and loss graph for the RMSprop optimised model.

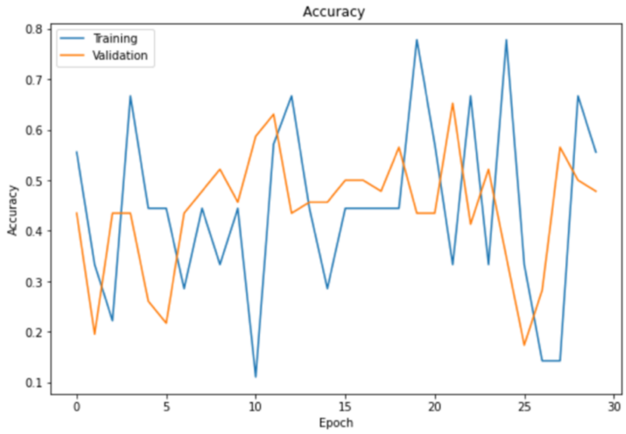

Figure 7: Accuracy graph for the RMSprop optimised model

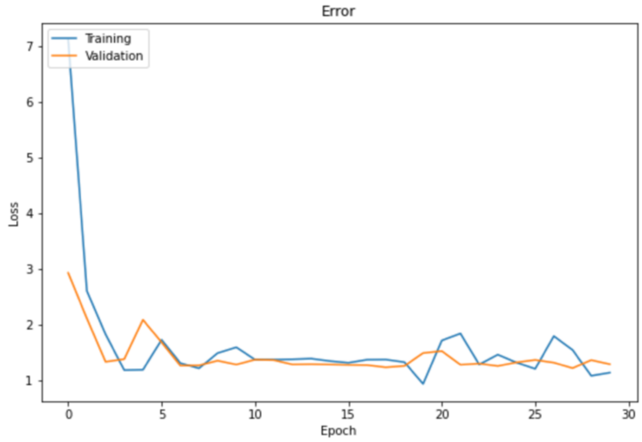

Figure 8: Loss graph for the RMSprop optimised model

From these graphs, it is clear to see how volatile the accuracy graph is during the training. The highest accuracy achieved during training was 100% but that only occurred once. The average accuracy was 86%. The error graph is definitely more stabilised, maintaining lower errors as the epochs go on.

Unfortunately, the visualisations of the prediction model appear to be largely broken. Despite being provided the 4 different classifications, it does not appear to predict any class aside from “noplastic” and “otherplastic”. This can be seen in Figure 9 below. However, it seems to somewhat accurately classify these images.

Figure 9: Prediction visualisation of the plastic waste data