IE4483 Artificial Intelligence and Data Mining Project

Project: Dogs vs. Cats

Group Members: Riya Anadkat (N2303616L), Ben Cheong (U2022838F), Isaac Tay (U2021546K)

Contributions:

Riya Anadkat	Ben Cheong	Isaac Tay
● CIFAR data preprocessing and augmentation ● CIFAR dataset model exploration and inference ● CIFAR model training and validation visualization	● Cats and dogs data preprocessing and inference ● MobileNet exploration and inference ● MobileNet model training and validation visualization	● CIFAR dataset model exploration and inference ● Resnet50 model exploration and inference ● ResNet50 model training and validation visualization

Report Questions

a) State the amount of image data that you used to form your training set and testing set. Describe the data pre-processing procedures and image augmentations (if any).

We have used the entire training and validation sets of (10000 training images and 2500 validation images). The image augmentations we used were scaling to normalize the pixel values between 0 and 1, randomly rotating the images, applying shearing transformations, zooming the images, and randomly flipping the images.

b) Select or build at least one machine learning model (e.g., CNN + linear layer, etc.) to construct your classifier. Clearly describe the model you use, including a figure of model architecture, the input and output dimensions, structure of the model, loss function(s), training strategy, etc.
Include your code and instructions on how to run the code. If non-deterministic method is used, ensure reproducibility of your results by averaging the results over multiple runs, cross-validation, fixing random seed, etc.

Training Strategy:
We used CNN to construct our classifier. For this problem, we experimented with multiple methods. First, we tried our self-made model. For this model, we used 3 convolution layers, the ReLU function and 2D max pooling is performed after every convolutional layer. This solution produced a validation accuracy of 89%. Then, for our second model we used ResNet50 as the base model, which has 50 layers, along with additional layers. This did not produce a suitable validation accuracy and it performed worse than Model 1, with an accuracy of 68%. Our final and most successful model uses MobileNet as the base model, which includes 53 convolutions layers, along with 2 additional layers. We use GlobalAveragePooling2D to reduce the spatial dimensions of the data to 1x1 and Dense to produce a single output.

Model Architecture:
Input dimensions = (224, 224, 3) This is the dimensions of the input image that MobileNet model accepts
Output dimensions = (1,) The output is a one-dimensional array with one element. The value will be in the range [0, 1], which represents the probability of the input image belonging to the 'dog' class.

Figure 1a: Model 1 Structure - Self Made

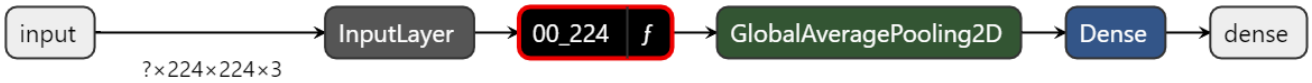

Figure 1b: Model 3 Structure - MobileNet

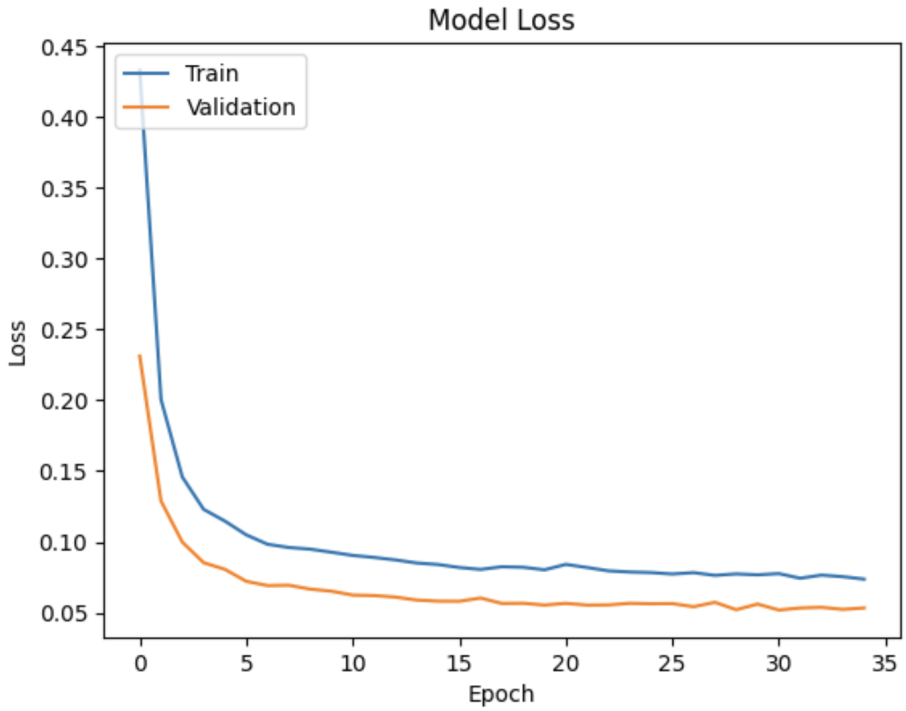

Figure 2: Loss Function of mobilenet

Code Instructions: Our code can be run on Google Colab. The Link is IE4483.ipynb

c) Discuss how you consider and determine the parameters (e.g., learning rate, etc.) / settings of your model as well as your reasons of doing so.

The learning rate we used for our MobileNet was 0.0001. We chose a low learning rate to give the model more training time for accurate results. Through experimentation, we realized that MobileNet-v2 gave us optimal accuracy, which is why the model has 53 layers. Having so many layers increases the complexity of the model, thus increasing the accuracy as well since each layer increases the model's ability to recognize patterns in the images.

d) Report the classification accuracy on validation set. Apply the classifier(s) built to the test set. Submit the "submission.csv" with the results you obtained.

The validation set reached an accuracy of 98% by convergence.

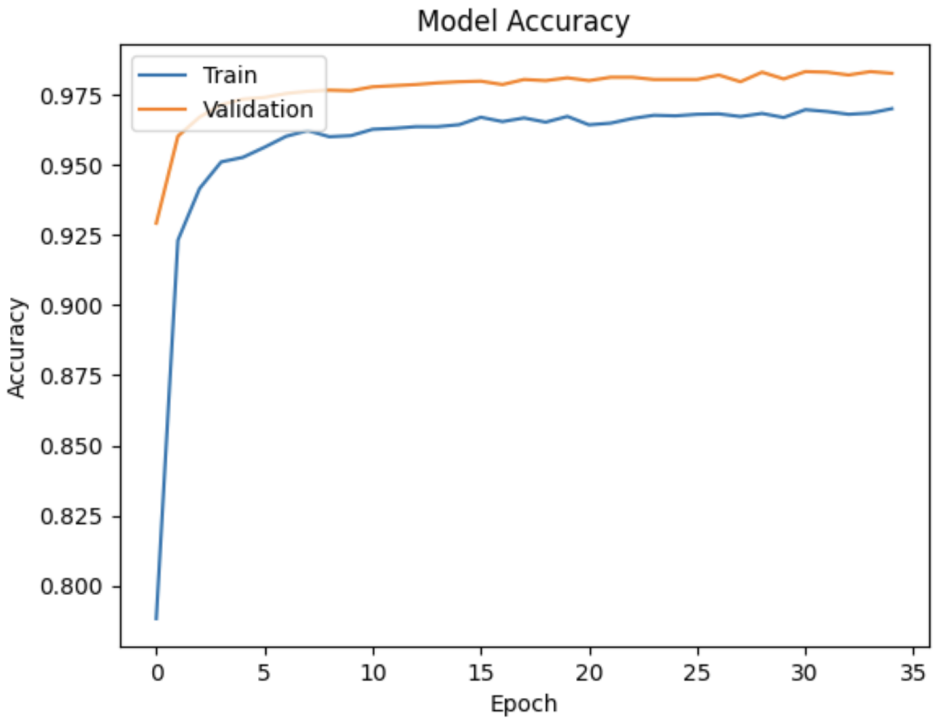

Figure 3: Model Accuracy of mobilenet

The submission.csv for our results can be viewed here.

e) Analyse some correctly and incorrectly classified (if any) samples in the test set. Select 1-2 cases to discuss the strength and weakness of the model.

Most of our samples are correct. Our model is able to classify different breeds and colors of the dogs and cats correctly. Image number 97 is quite ambiguous since all that can be seen is a black silhouette of a dog, but our model was still able to correctly predict it. Our model's strength is that it can recognise edges, shapes, colors, and textures.

An incorrectly classified sample in the test set is number 16. It was incorrectly labeled as a dog by our model. The image shows the top view of a cat, which is an outlier in our test data set.

Since most of the training images are the close up and eye level, it is likely that the model did not recognise the top-view of the cat. Thus, a weakness of our model is that it is not able to learn patterns effectively when there is limited data for a subset of a category (e.g., top-view of a cat). Since most images that are top-view are of dogs, this could indicate that our model is very sensitive to our training data, thus this learning error is due to variance.



f) Discuss how different choice of models and data processing may affect the project in terms of accuracy on validation set.

Having a model that is too complex e.g. using too many layers, may cause the validation set accuracy to decrease due to overfitting. This means that the accuracy of the training set may be high, but the accuracy of the validation set is much lower/not as accurate. This is because the model is trained to fit the training data strictly, but it is not able to generalize. Using a dropout layer can prevent overfitting since it sets input elements to zero with a given probability. Also, if the amount of training data is too small, it may not contain enough data samples to represent all data values, thus causing poor validation accuracy. Having image augmentation in data processing acts as a form of regularization by adding noise to the training process. This can help prevent the model from learning patterns that are specific to the training data but do not generalize well to new data, hence, potentially increasing the accuracy on the validation dataset.

g) Apply and improve your classification algorithm to a multi-category image classification problem for Cifar-10 dataset (https://www.cs.toronto.edu/~kriz/cifar.html). Describe details about the dataset and classification problem that you are solving. Explain how your algorithm can tackle this problem, and what changes you make comparing to solving Dogs vs. Cats problem. Report your results for the testing set of Cifar-10.

Training Strategy:

The Cifar-10 dataset contains 10 classes with 6000 images per class. Our binary classification algorithm for 'Dogs vs. Cats' is not the most effective for a multi-category image problem. Instead, for this problem, we chose to create our own layers. We used 9 convolution layers with ReLu activation, and batch normalization. We used 3 dropout layers to prevent overfitting and flatten for converting the multi-dimensional network to a 1-dimensional vector. We were able to reach a validation accuracy of 89% using a learning rate of 0.001 and 413 epochs. Our new model for the Cifar-10 dataset has a lower complexity compared to our previous model because we wanted to limit learning error due to variance which increases as model complexity increases.

Model Architecture:

Input dimensions = (32, 32, 3) This is the dimensions of the input image given from the Cifar-10 dataset
Output dimensions = (None,10) 10 is the number of neurons in the Dense layer. Each element represents the probability of the input belonging to one of the 10 classes.

Layer (type)	Output Shape	Param #
input_3 (InputLayer)	[(None, 32, 32, 3)]	0
conv2d_9 (Conv2D)	(None, 32, 32, 64)	256
batch_normalization_17 (BatchNormalization)	(None, 32, 32, 64)	256
activation_9 (Activation)	(None, 32, 32, 64)	0
conv2d_10 (Conv2D)	(None, 30, 30, 64)	36928
batch_normalization_18 (BatchNormalization)	(None, 30, 30, 64)	256
activation_10 (Activation)	(None, 30, 30, 64)	0
conv2d_11 (Conv2D)	(None, 26, 26, 64)	102464
batch_normalization_19 (BatchNormalization)	(None, 26, 26, 64)	256
activation_11 (Activation)	(None, 26, 26, 64)	0
dropout_9 (Dropout)	(None, 26, 26, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 13, 13, 64)	0
conv2d_12 (Conv2D)	(None, 13, 13, 128)	8320
batch_normalization_20 (BatchNormalization)	(None, 13, 13, 128)	512
activation_12 (Activation)	(None, 13, 13, 128)	0
conv2d_13 (Conv2D)	(None, 11, 11, 128)	147584
batch_normalization_21 (BatchNormalization)	(None, 11, 11, 128)	512
activation_13 (Activation)	(None, 11, 11, 128)	0
conv2d_14 (Conv2D)	(None, 7, 7, 128)	409728
batch_normalization_22 (BatchNormalization)	(None, 7, 7, 128)	512
activation_14 (Activation)	(None, 7, 7, 128)	0
dropout_10 (Dropout)	(None, 7, 7, 128)	0
conv2d_15 (Conv2D)	(None, 7, 7, 256)	33024
batch_normalization_23 (BatchNormalization)	(None, 7, 7, 256)	1024
activation_15 (Activation)	(None, 7, 7, 256)	0
conv2d_16 (Conv2D)	(None, 5, 5, 256)	590080
batch_normalization_24 (BatchNormalization)	(None, 5, 5, 256)	1024
activation_16 (Activation)	(None, 5, 5, 256)	0
conv2d_17 (Conv2D)	(None, 1, 1, 256)	1638656
batch_normalization_25 (BatchNormalization)	(None, 1, 1, 256)	1024
activation_17 (Activation)	(None, 1, 1, 256)	0
dropout_11 (Dropout)	(None, 1, 1, 256)	0
flatten_3 (Flatten)	(None, 256)	0
dense_9 (Dense)	(None, 10)	2570

Figure 4: Model Structure for Cifar-10 classification

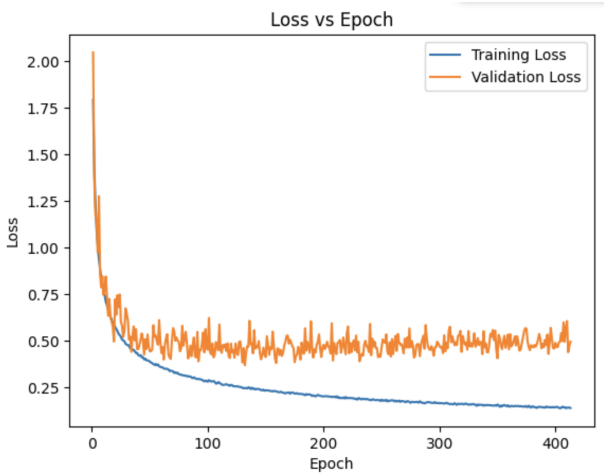

Figure 5: Loss function for Cifar-10 classification

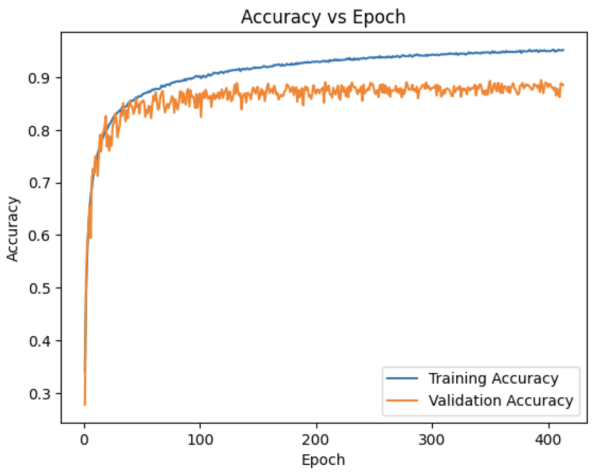

Figure 6: Model Accuracy for Cifar-10 classification

h) Train the classifier for (g), while some of the classes in Cifar-10 training dataset contains much fewer labelled data. How can you improve your algorithm to tackle the data unbalancing issue? Describe and justify at least 2 approaches you use.

One way to improve the algorithm to tackle the data unbalancing issue is image augmentation. Image augmentation helps to artificially expand the data set with a few data samples. It can help increase the diversity of our training sets by adding random transformations to our existing images. Adding variations to the data set can improve the generalizability of the model.
Another way to improve the data unbalancing issue is to lower the learning rate and increase the number of epochs. Lowering the learning rate means that there are more data points for the algorithm to learn. It would be a good idea to increase the number of epochs because the model may take longer to reach convergence since it needs more time to learn from the training data.

References

1. TensorFlow. (n.d.). tf.keras.applications.ResNet50V2. Retrieved [15/11/23] from, https://www.tensorflow.org/api_docs/python/tf/keras/applications/resnet_v2/ResNet50V2

2. TensorFlow Authors (n.d.). Convolutional Neural Network (CNN). Retrieved [15/11/23] from, https://www.tensorflow.org/tutorials/images/cnn