1. DATASET AND DATA CLEANING

Stanford Dogs Datasets (*https://www.kaggle.com/jessicali9530/stanford-dogs-dataset*)

The Stanford Dogs dataset contains images of 120 breeds of dogs from around the world. This dataset has been built using images from ImageNet. It was originally collected for fine-grain image categorization, a challenging problem as certain dog breeds have near identical features or differ in colour and age.

- Number of class labels : 120
- Number of images : 20,580
- ~ 150 images per class

The images were provided as JPG files. They were divided into sub-directories, with each folder containing all the images belonging to a specific class.

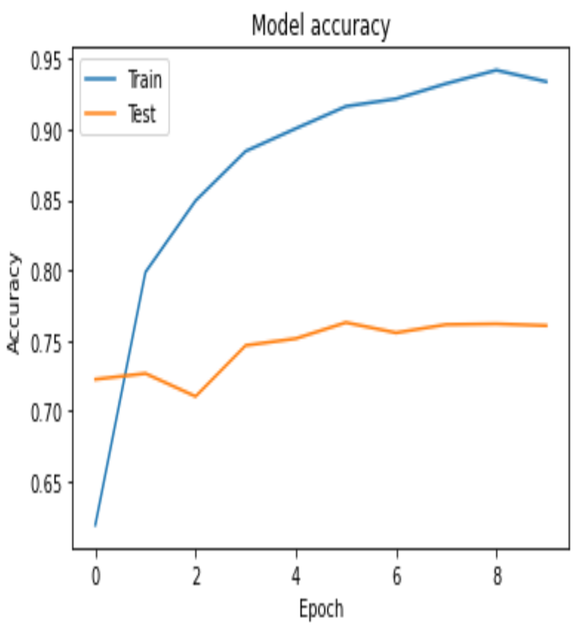
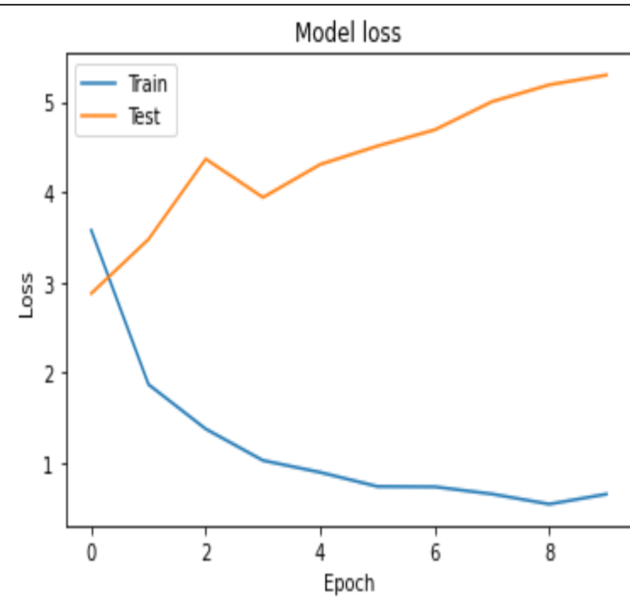
- `tf.keras.preprocessing.image_dataset_from_directory` was used to read the images into the training and testing datasets, with a an 80-20 train-test split
- The images were resized to 224 x 224
- The batch size was set to 48 for training images, and 25 for testing images
- The class labels were updated to contain only the names of the dog breeds, extra characters were removed.
- The data was standardized to be in the [0, 1] range using `layers.experimental.preprocessing.Rescaling(1./255)`.

2. TRANSFER LEARNING MODEL

The selected pre-trained model was ResNet50. More precisely, resnet_50/feature_vector *(https://tfhub.dev/tensorflow/resnet_50/feature_vector/1)* which produces feature vectors of images. The weights in this model have been obtained by training on the 'ImageNet' dataset.

3. EXPERIMENT LOG

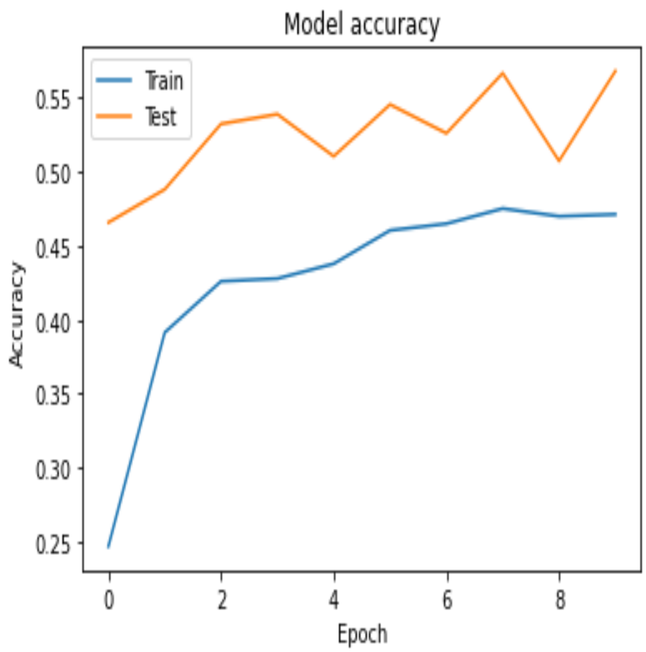
EXPERIMENT 1:

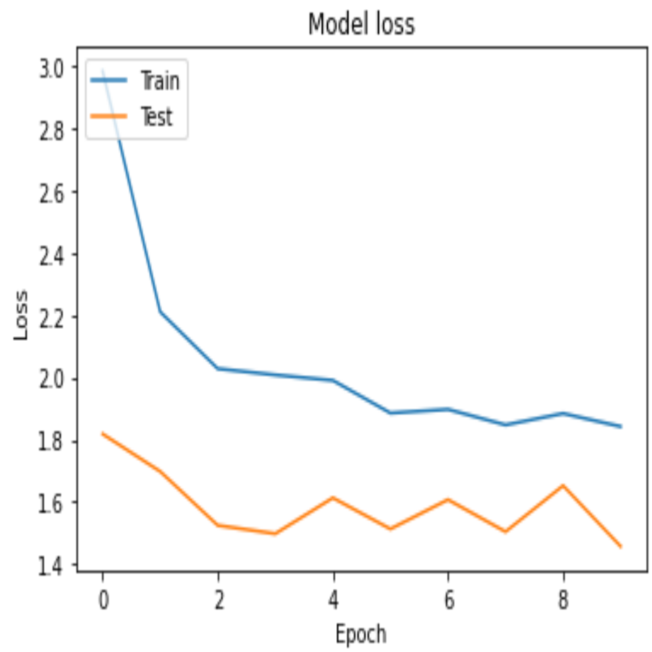
Number of Layers	1
Layer 1 Type	Dense
Layer 1 Activation Function	Linear
Epochs	10
Learning Rate	0.01
Training Accuracy	0.9339 → 93%
Test Accuracy	0.7606 → 76%
Training Loss	0.6569
Test Loss	5.2981
	

EXPERIMENT 2:

Number of Layers	3
Layer 1 Type	Dense
Layer 1 Activation Function	ReLu
Layer 1 Dropout Rate	20%
Layer 2 Type	Dense
Layer 2 Activation Function	ReLu
Layer 3 Type	Dense
Layer 3 Activation Function	Linear

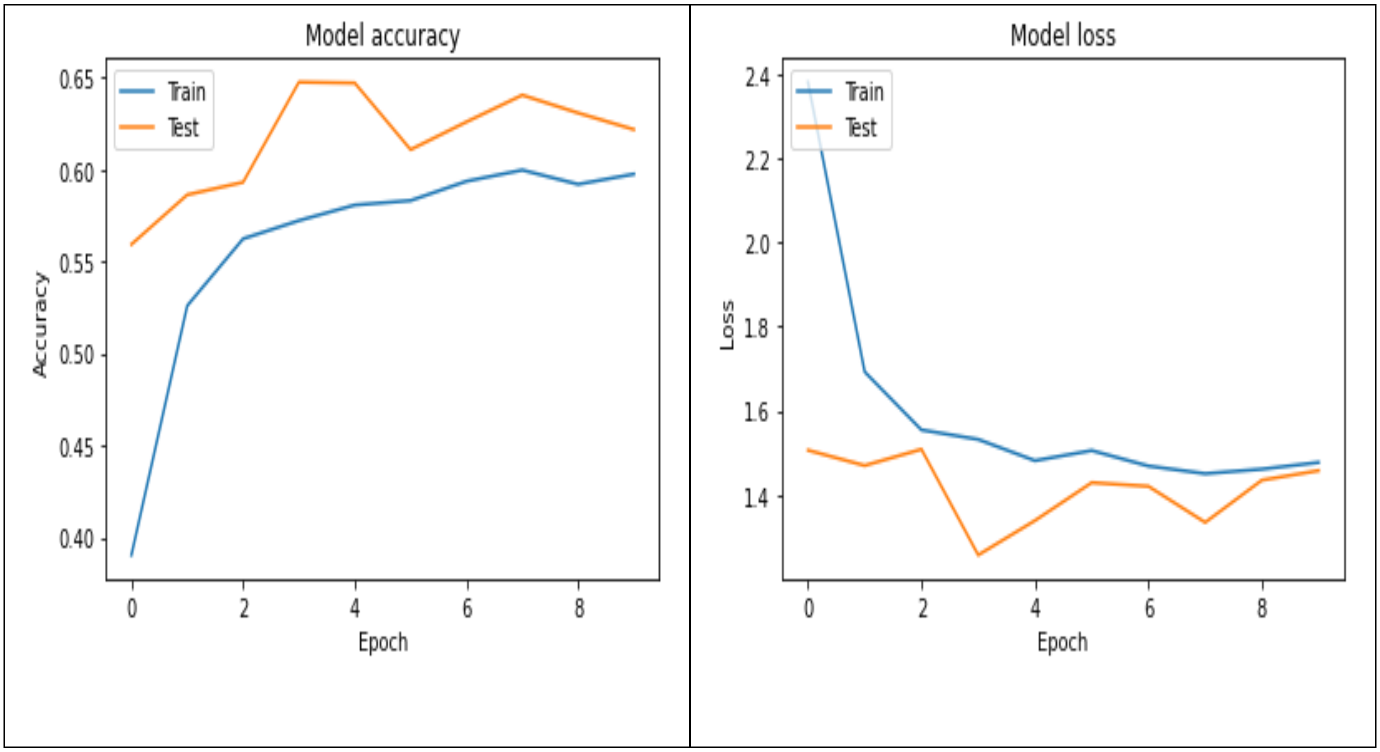
Epochs	10
Learning Rate	0.01
Training Accuracy	0.4712 → 47%
Test Accuracy	0.5680 → 56%
Training Loss	1.8421
Test Loss	1.4566





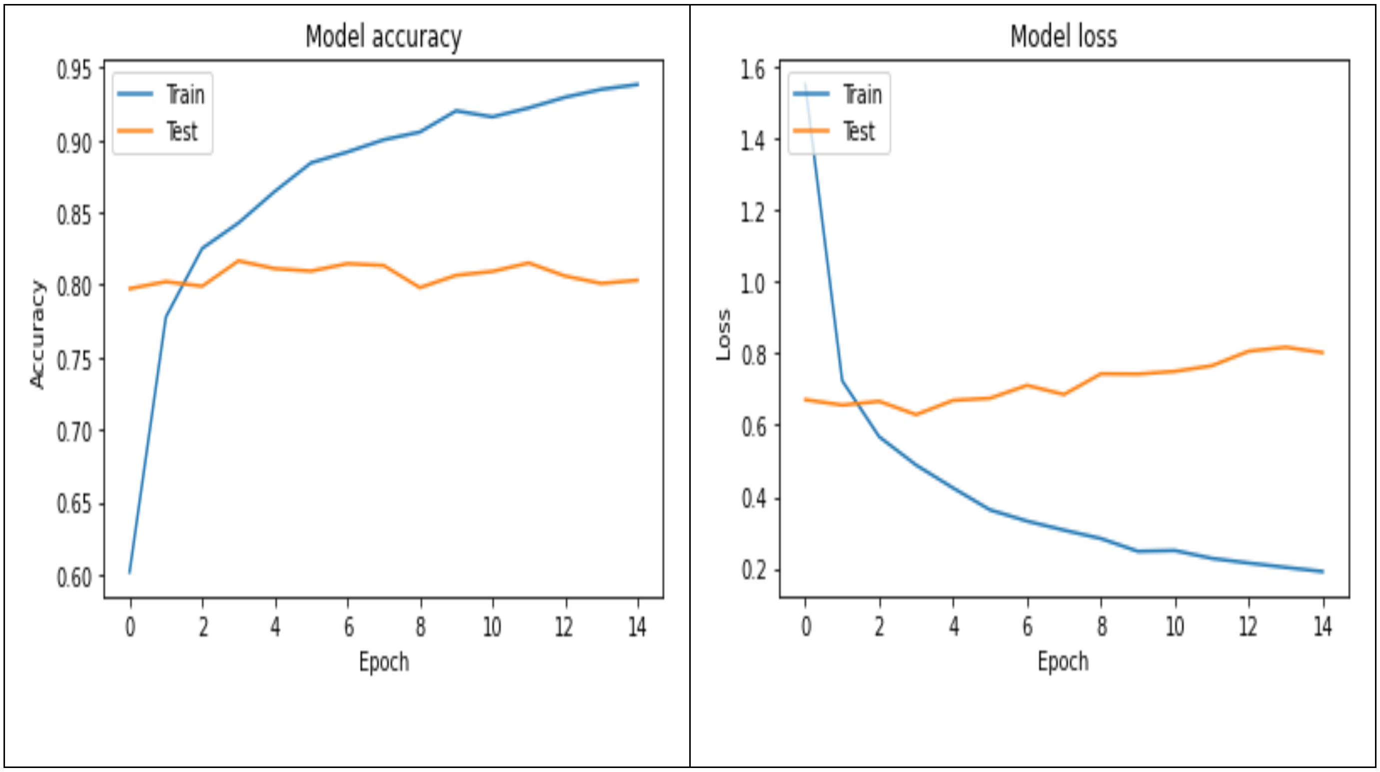
EXPERIMENT 3:

Data Augmentation	• Random horizontal flipping • Random rotation • Random zoom
Number of Layers	3
Layer 1 Type	Dense
Layer 1 Activation Function	ReLu
Layer 2 Type	Dense
Layer 2 Activation Function	ReLu
Layer 3 Type	Dense
Layer 3 Activation Function	Linear
Epochs	10
Learning Rate	0.01
Training Accuracy	0.5974 → 59%
Test Accuracy	0.6217 → 62%
Training Loss	1.4768
Test Loss	1.4573



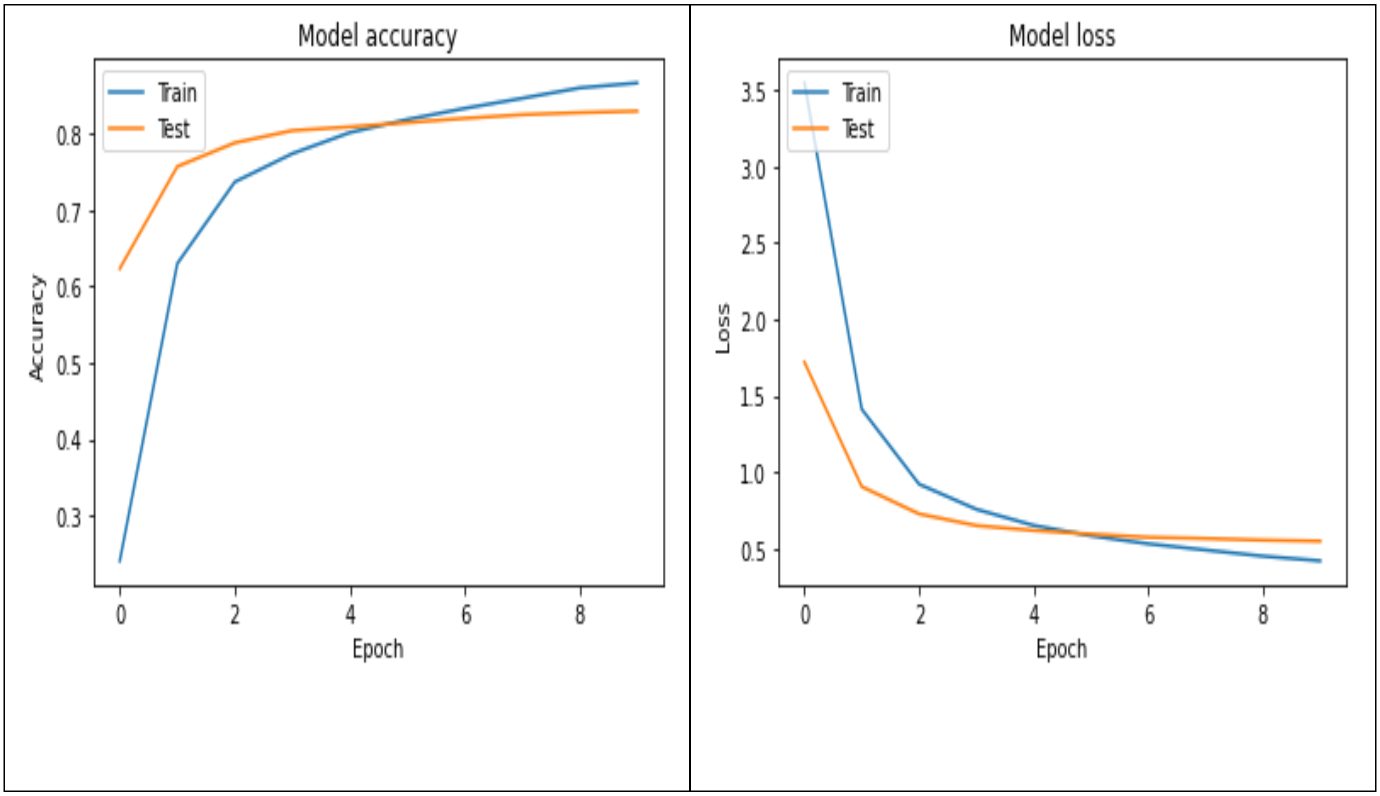
EXPERIMENT 4:

Data Augmentation	• Random horizontal flipping • Random rotation • Random zoom
Number of Layers	1
Layer 1 Type	Dense
Layer 1 Activation Function	Linear
Epochs	15
Learning Rate	0.001
Training Accuracy	0.9382 → 93%
Test Accuracy	0.8032 → 80%
Training Loss	0.1910
Test Loss	0.8019



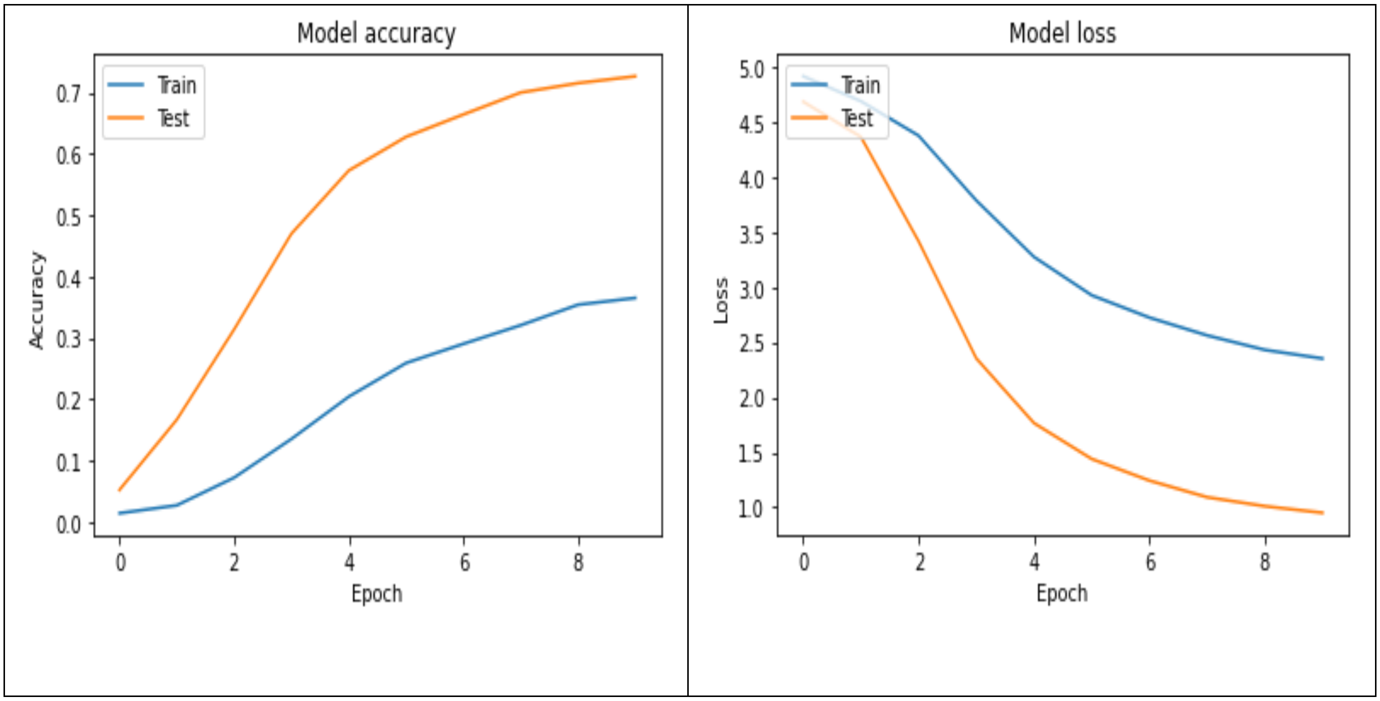
EXPERIMENT 5:

Data Augmentation	Random horizontal flipping
Number of Layers	3
Layer 1 Type	Dense
Layer 1 Activation Function	ReLu
Layer 1 Dropout Rate	20%
Layer 2 Type	Dense
Layer 2 Activation Function	ReLu
Layer 3 Type	Dense
Layer 3 Activation Function	Linear
Epochs	10
Learning Rate	0.0001
Training Accuracy	0.8671 → 86%
Test Accuracy	0.8301 → 83%
Training Loss	0.4189
Test Loss	0.5477



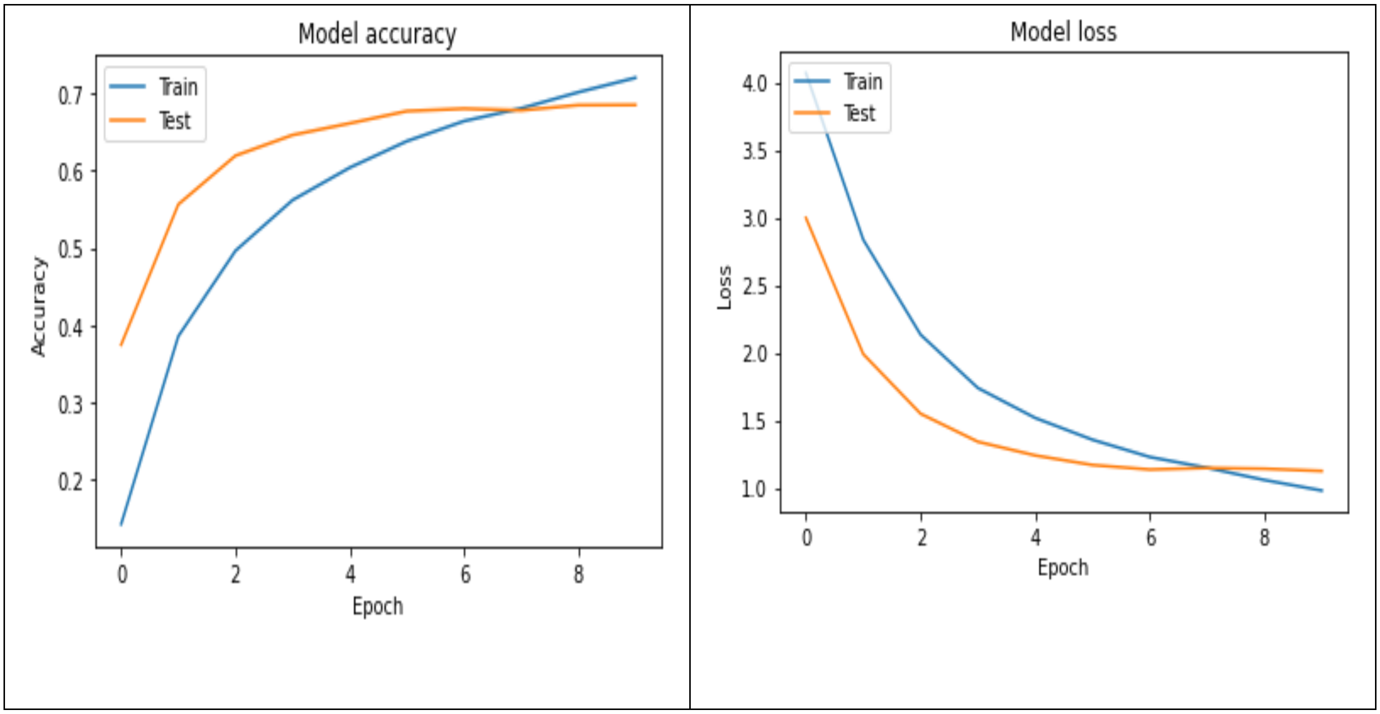
EXPERIMENT 6:

Data Augmentation	• Random horizontal flipping • Random rotation • Random zoom
Number of Layers	3
ResNet Dropout Rate	50%
Layer 1 Type	Dense
Layer 1 Activation Function	ReLu
Layer 1 Dropout Rate	50%
Layer 2 Type	Dense
Layer 2 Activation Function	ReLu
Layer 3 Type	Dense
Layer 3 Activation Function	Linear
Epochs	10
Learning Rate	0.0001
Training Accuracy	0.3650 → 36%
Test Accuracy	0.7261 → 72%
Training Loss	2.3512
Test Loss	0.9501



EXPERIMENT 7:

Data Augmentation	• Random rotation • Random zoom
Number of Layers	6
Layer 1 Type	Convolutional
Layer 1 Activation Function	ReLu
Layer 1 Filter Size	5 x 5
Layer 1 Filter Count	32
Layer 4 Type	Dense
Layer 4 Activation Function	ReLu
Layer 4 Dropout Rate	20%
Layer 5 Type	Dense
Layer 5 Activation Function	ReLu
Layer 6 Type	Dense
Layer 6 Activation Function	Linear
Epochs	10
Learning Rate	0.0001
Training Accuracy	0.7196 → 71%
Test Accuracy	0.6846 → 68%
Training Loss	0.9792
Test Loss	1.1238


Model accuracy
Train
Test
Accuracy
Epoch
0.7
0.6
0.5
0.4
0.3
0.2
0
2
4
6
8
Model loss
Train
Test
Loss
Epoch
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0
2
4
6
8

4. TEST RESULTS

EXPERIMENT 1:

Figure 1: experiment 1 model predictions vs. true labels for 25 test images

EXPERIMENT 2:

Figure 2: experiment 2 model predictions vs. true labels for 25 test images

EXPERIMENT 3:

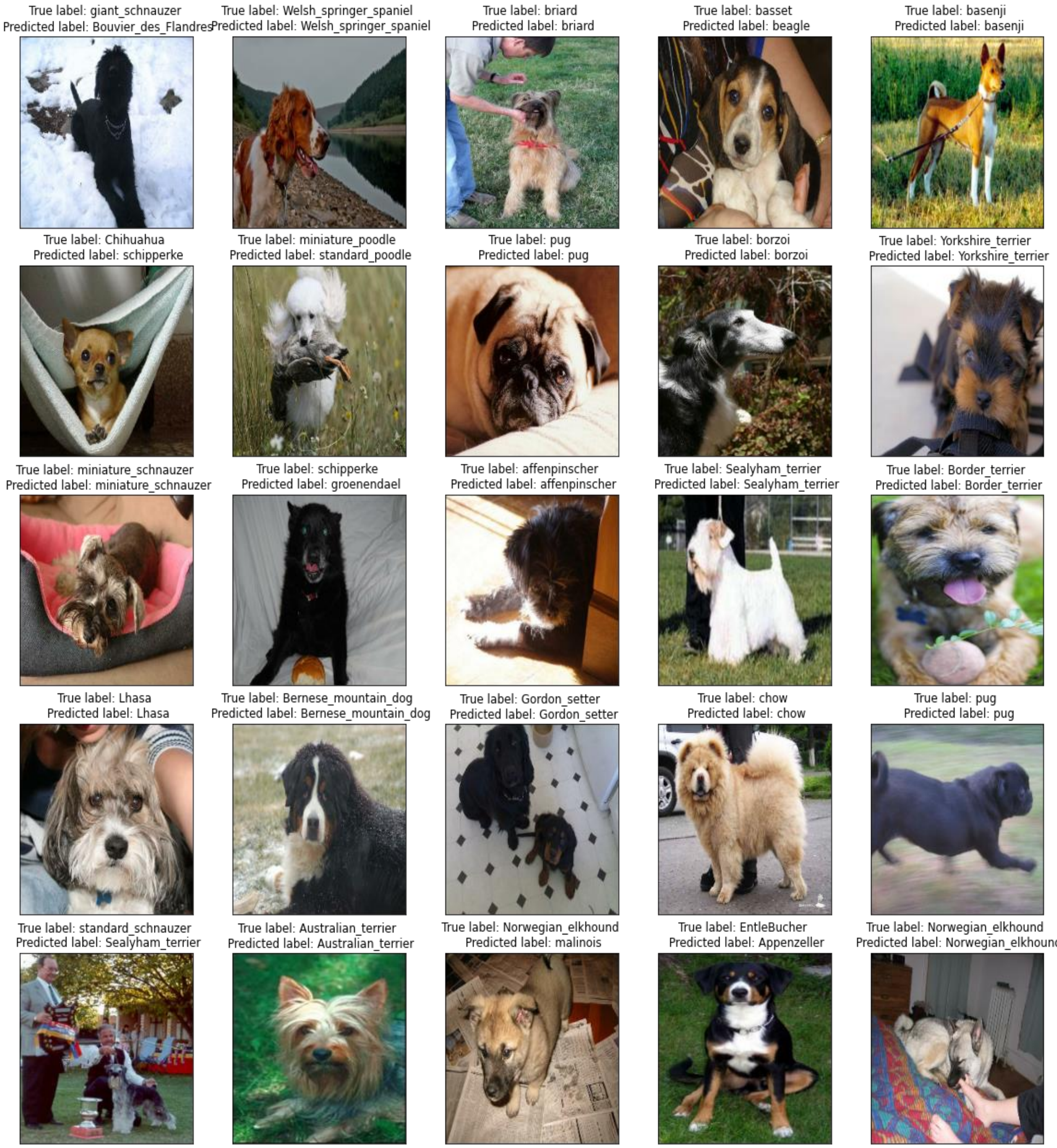


Figure 3: experiment 3 model predictions vs. true labels for 25 test images

EXPERIMENT 4:

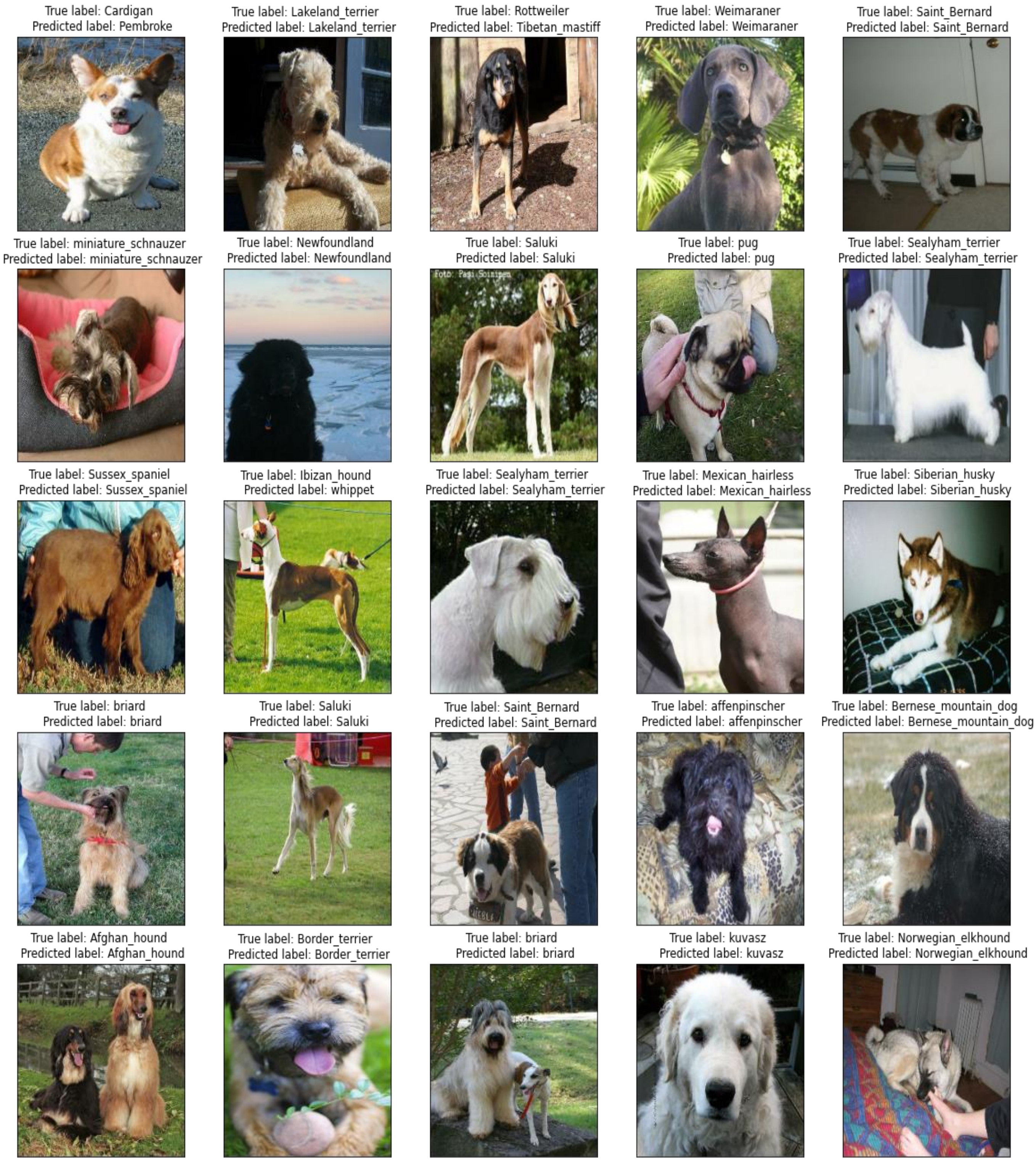


Figure 4: experiment 4 model predictions vs. true labels for 25 test images

EXPERIMENT 5:

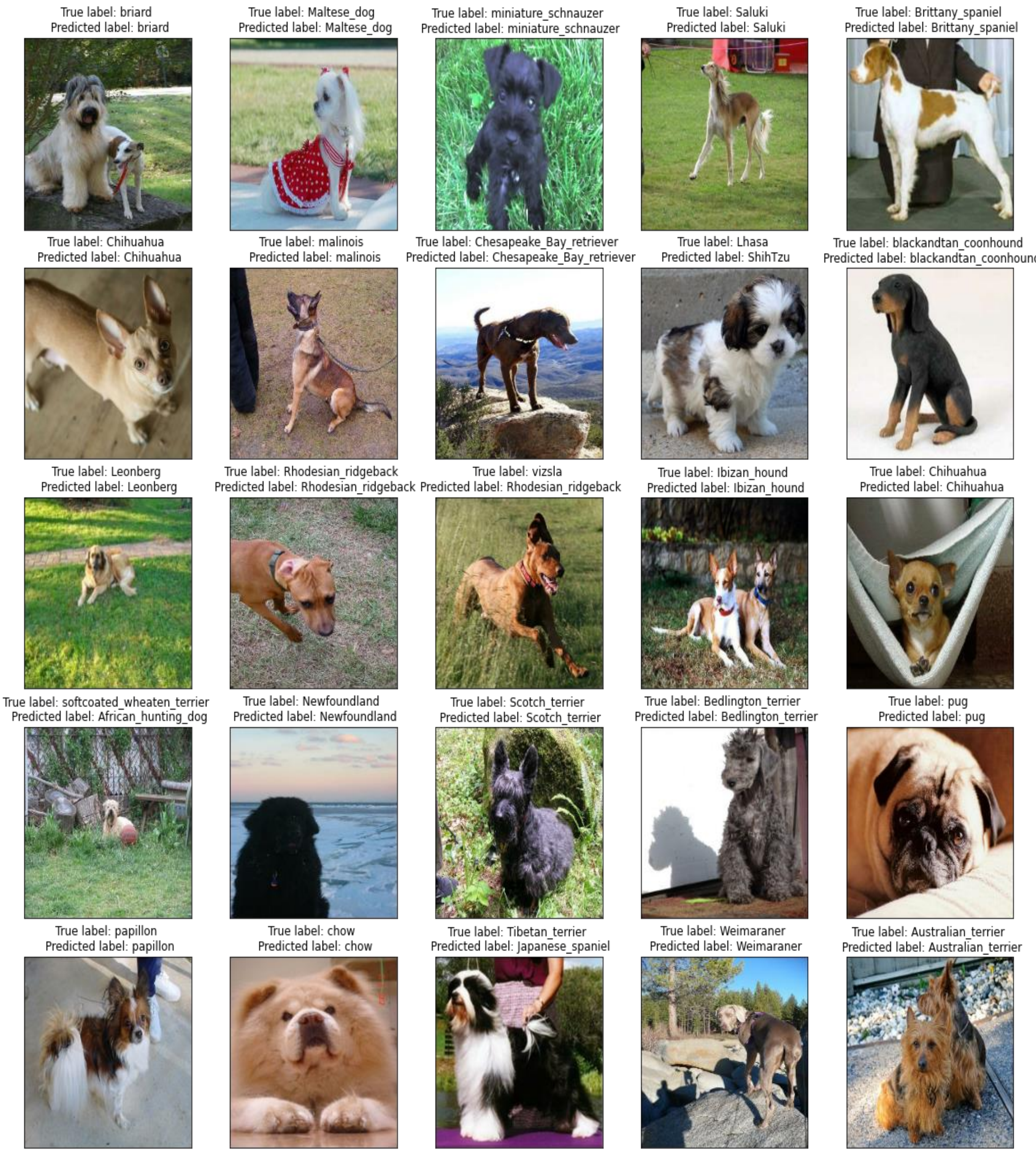


Figure 5: experiment 5 model predictions vs. true labels for 25 test images

EXPERIMENT 6:

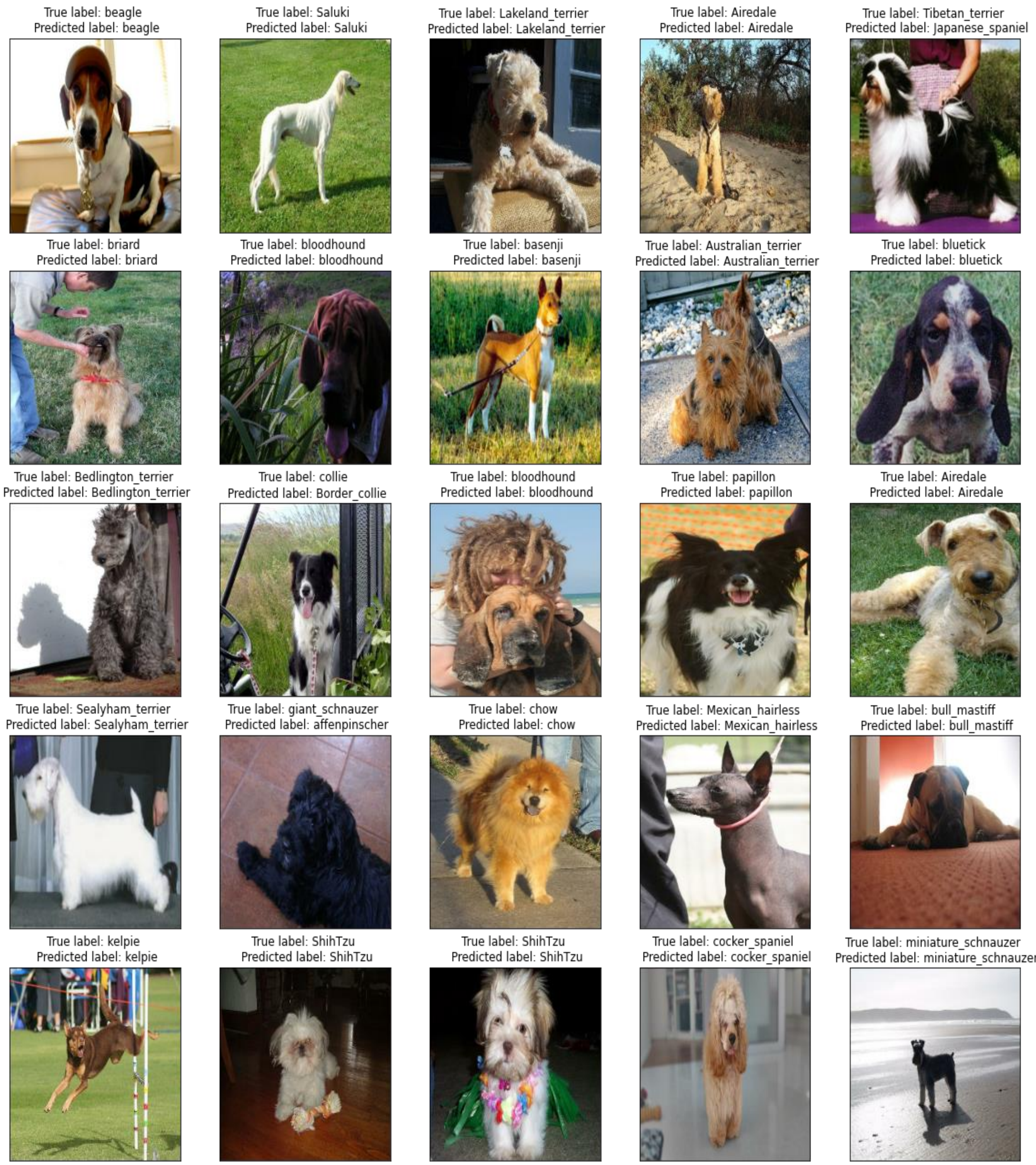


Figure 6: experiment 6 model predictions vs. true labels for 25 test images

EXPERIMENT 7:

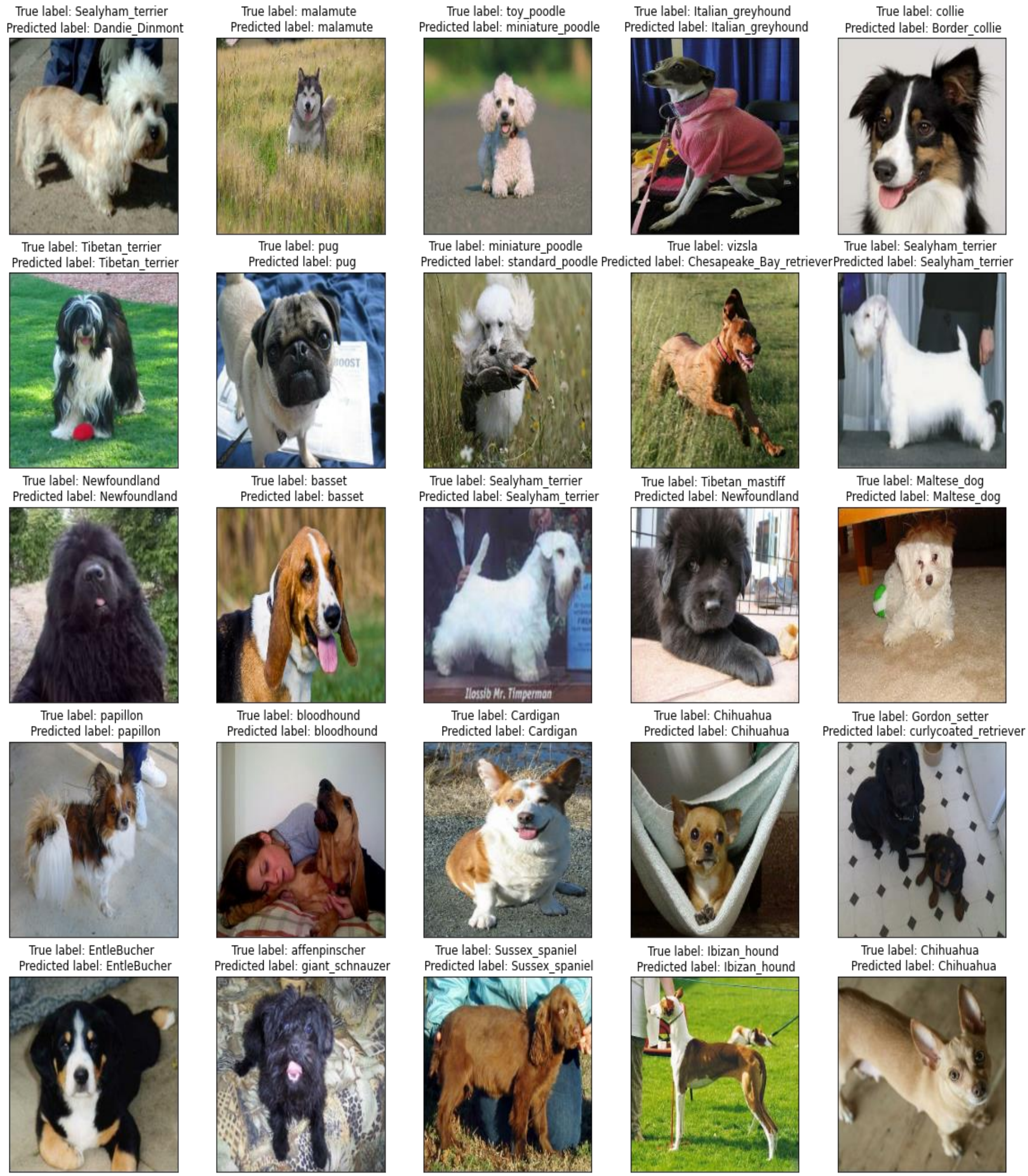


Figure 7: experiment 7 model predictions vs. true labels for 25 test images

5. INTERPRETATION OF RESULTS

In the 1st experiment, we added a single output layer after the pre-trained ResNet model. The resulting training accuracy of 93% and testing accuracy of 76%, along with a high test loss of 5.29, indicated overfitting had occurred.

To prevent overfitting, we used a dropout rate of 20% in experiment 2, and data augmentation in experiment 3, but the resulting training and testing accuracy was unsatisfactory. This low accuracy suggested that another parameter needed to be tuned.

Based on the results of the previous experiments, in the 4th experiment, we updated the learning rate from 0.01 to 0.001, and also utilized data augmentation to prevent overfitting. In addition, 15 epochs were run instead of 10. This produced a 93% training accuracy and an 80% test accuracy, signalling that overfitting may have occurred, possibly due to the large number of epochs.

In the next experiment, we used both data augmentation and dropout to prevent overfitting, and further lowered the learning rate to 0.0001. Besides this, we ran the model for only 10 epochs. This experiment achieved a training accuracy of 86%, and the highest yet test accuracy of 83%.

We tuned the dropout in the 6th experiment, increasing it to 50%, but this resulted in a sharp drop in training accuracy to 36%, demonstrating that with such a high dropout rate, additional epochs may be needed to achieve satisfactory training accuracy.

Finally, in the last experiment,, we tuned the ResNet itself, adding 1 convolutional layer with 32 (5x5) filters, as well as a Max Pooling layer. In this model as well, the images were augmented, and there was a dropout rate of 20%. However, 68% was the test accuracy obtained, suggesting that for the problem of dog-breed classification, no fine-tuning of the ResNet model is needed.

6. CONCLUSION

We concluded that the model from experiment 5 produced the best results. This model had, in addition to the 50 ResNet layers, 2 dense hidden layers - with 256 and 128 perceptrons respectively - both using the ReLu activation function.
In order to prevent overfitting, this model employed data augmentation in the form of random horizontal flipping, along with a dropout rate of 20% after the first hidden layer.
With a learning rate of 0.0001, and 10 epochs, this model was able to achieve a test accuracy of 83%. Out of the 25 test images, it correctly classified 21 dog breeds.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
random_flip_1 (RandomFlip)   (None, 224, 224, 3)       0
_________________________________________________________________
keras_layer_1 (KerasLayer)   (None, 2048)              23561152
_________________________________________________________________
dense_1 (Dense)              (None, 256)               524544
_________________________________________________________________
dropout (Dropout)            (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_3 (Dense)              (None, 120)               15480
=================================================================
Total params: 24,134,072
Trainable params: 572,920
Non-trainable params: 23,561,152
```

Figure 8: selected model architecture