BIOS 740 Computing HW 3 – Code/Figures Appendix

CNN Model Parameters

```
Loaded Tensorflow version 2.0.0
2021-11-17 23:19:16.626441: I tensorflow/core/platform/cpu_feature_guard.cc:145] This TensorFlow binary is optimized with Intel(R)
MKL-DNN to use the following CPU instructions in performance critical operations:  SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in non-MKL-DNN operations, rebuild TensorFlow with the appropriate compiler flags.
2021-11-17 23:19:16.626833: I tensorflow/core/common_runtime/process_util.cc:115] Creating new thread pool with default inter op
setting: 4. Tune using inter_op_parallelism_threads for best performance.
the `lr` argument has been renamed to `learning_rate`.Model: "sequential"
_____
Layer (type)                         Output Shape                      Param #
=================================================================
conv2d_2 (Conv2D)                    (None, 148, 148, 16)              448
_____
max_pooling2d_2 (MaxPooling2D)       (None, 74, 74, 16)                0
_____
conv2d_1 (Conv2D)                    (None, 72, 72, 32)                4640
_____
max_pooling2d_1 (MaxPooling2D)       (None, 36, 36, 32)                0
_____
conv2d (Conv2D)                      (None, 34, 34, 64)                18496
_____
max_pooling2d (MaxPooling2D)         (None, 17, 17, 64)                0
_____
flatten (Flatten)                    (None, 18496)                     0
_____
dense_1 (Dense)                      (None, 512)                       9470464
_____
dense (Dense)                        (None, 1)                         513
=================================================================
Total params: 9,494,561
Trainable params: 9,494,561
Non-trainable params: 0
_____
```

Confirmation that generated batches have 250 images with 150x150 pixel dimensions, 3 RGB entries, and a 0 or 1 classifier for cat vs. dog

```
Found 24560 images belonging to 2 classes.
Found 5549 images belonging to 2 classes.
List of 2
 $ : num [1:250, 1:150, 1:150, 1:3] 0.247 0.173 0.137 0.231 0.549 ...
 $ : num [1:250(1d)] 1 1 0 1 0 1 1 1 1 0 ...
List of 2
 $ : num [1:250, 1:150, 1:150, 1:3] 0.475 0.624 0.549 0.678 0.227 ...
 $ : num [1:250(1d)] 0 1 1 0 1 1 0 0 0 0 ...
Time difference of 5.116195 secs
```

Epoch progress bars, time elapsed, training loss, training accuracy, validation loss, & validation accuracy

```
Epoch 1/5
50/50 [==============================] - 608s 12s/step - loss: 0.8641 - accuracy:
0.5747 - val_loss: 0.6411 - val_accuracy: 0.6040
Epoch 2/5
50/50 [==============================] - 619s 12s/step - loss: 0.6185 - accuracy:
0.6621 - val_loss: 0.5608 - val_accuracy: 0.7064
Epoch 3/5
50/50 [==============================] - 611s 12s/step - loss: 0.5986 - accuracy:
0.6927 - val_loss: 0.5252 - val_accuracy: 0.7536
Epoch 4/5
50/50 [==============================] - 601s 12s/step - loss: 0.5350 - accuracy:
0.7318 - val_loss: 0.4808 - val_accuracy: 0.7736
Epoch 5/5
50/50 [==============================] - 582s 12s/step - loss: 0.5064 - accuracy:
0.7498 - val_loss: 0.4631 - val_accuracy: 0.7750
```

5 Epoch overall time elapsed

```
> end.time <- Sys.time()
> time.elapsed <- end.time-start.time
> print(time.elapsed)
Time difference of 50.37726 mins
>
```

Validation loss & accuracy from optimal epoch

```
> #print out best loss and its corresponding accuracy
> epoch <- which.min(model_history$metrics$val_loss)
> loss <- round(model_history$metrics$val_loss[epoch],3)
> accuracy <- round(model_history$metrics$val_accuracy[epoch],3)
> print(loss)
[1] 0.463
> print(accuracy)
[1] 0.775
```

Plot of training & validation loss & accuracy for epochs 1-5