Transfer Learning Dog Breed Classification

Connor Kordes - JCK160130
Nathan Tompkins - NAT180002
University of Texas at Dallas
CS 4372

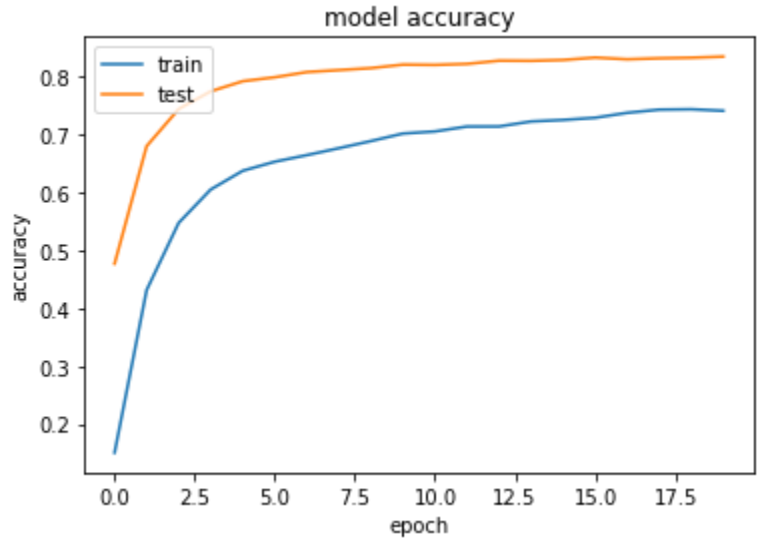
History Plots and Models

In all of the below imagenet2 is the base_model with include_top set to false.

Model 1:

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

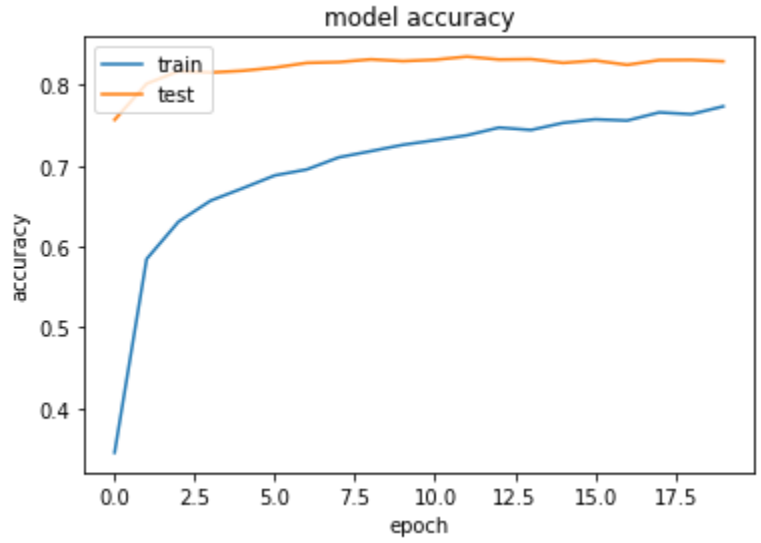
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



Model 2

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

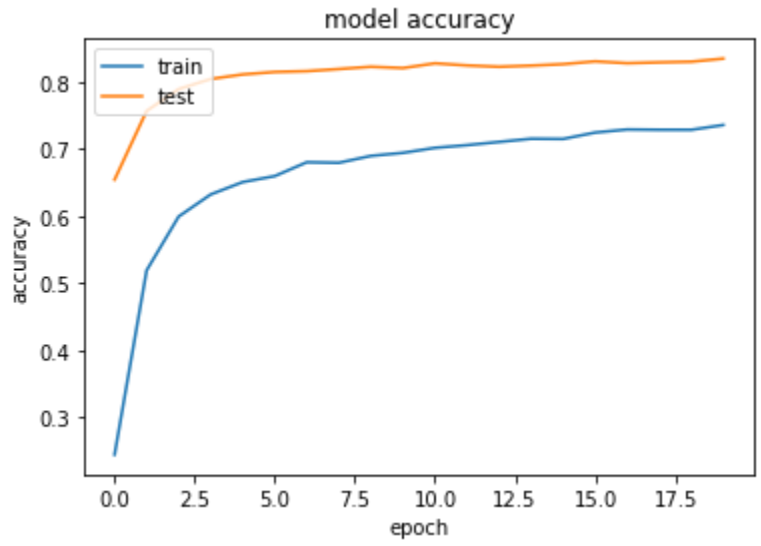
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



Model 3

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='sigmoid'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

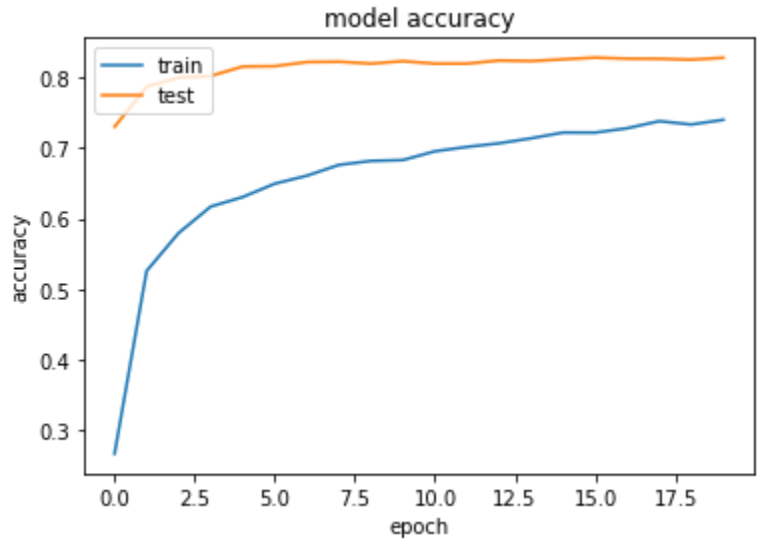
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



Model 4

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(500, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

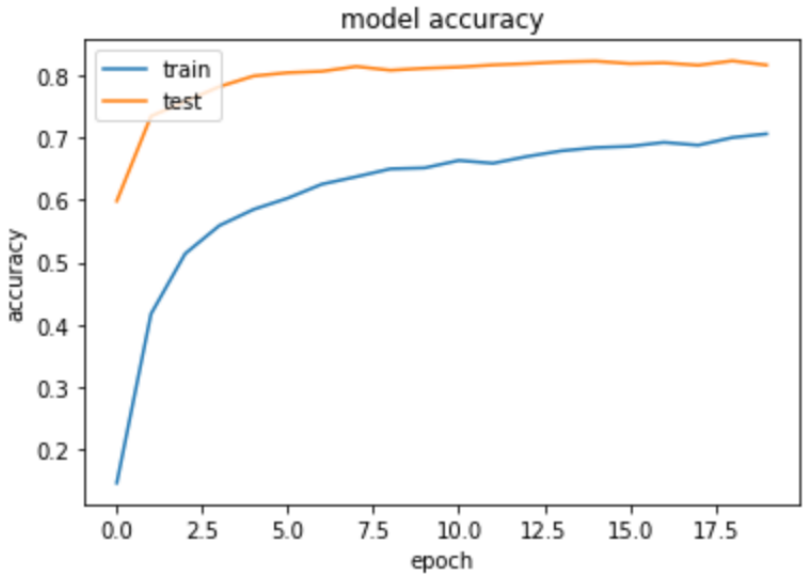
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



Model 5

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='sigmoid'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(500, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

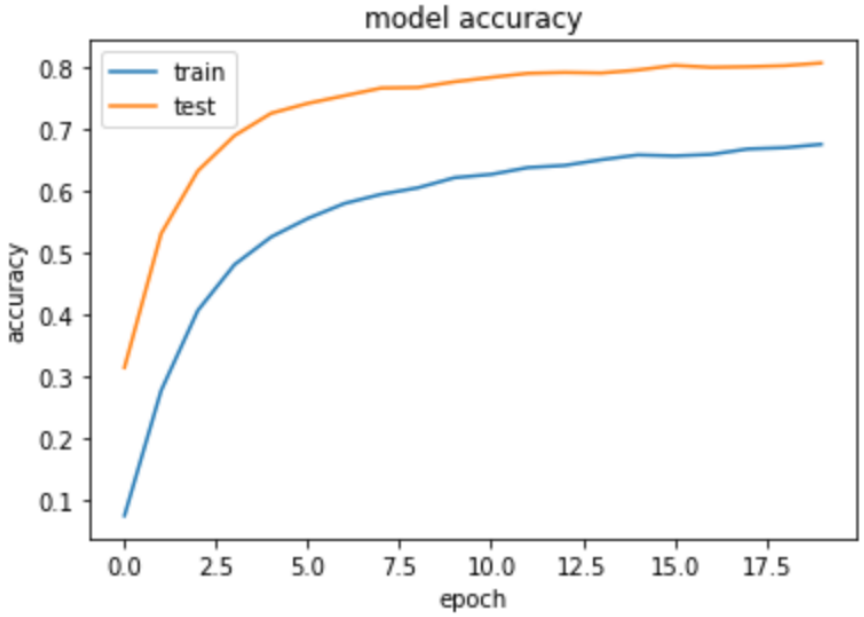
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



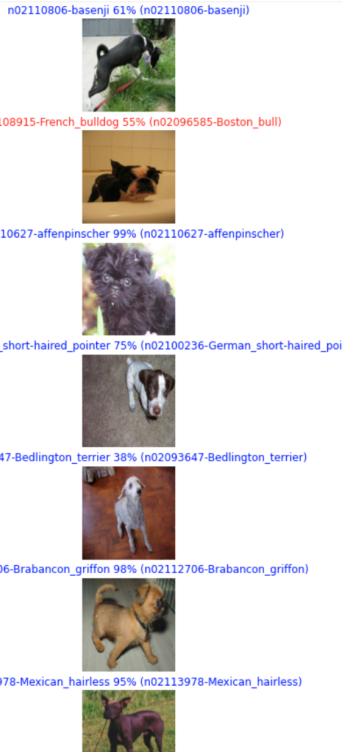
Model 6

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.RandomFlip('horizontal'))
model.add(tf.keras.layers.RandomRotation(0.2))
model.add(rescale)
model.add(base_model)
model.add(global_average_layer)
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='sigmoid'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1000, activation='sigmoid'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(120, activation='softmax'))
```

```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['accuracy'])
```



Sample Predictions of Model 1



n02092002-Scottish_deerhound 52% (n02092002-Scottish_deerhound)

n02093647-Bedlington_terrier 38% (n02093647-Bedlington_terrier)

n02091244-Ibizan_hound 98% (n02091244-Ibizan_hound)

n02112706-Brabancon_griffon 98% (n02112706-Brabancon_griffon)

n02086079-Pekinese 90% (n02086079-Pekinese)

n02113978-Mexican_hairless 95% (n02113978-Mexican_hairless)

n02107908-Appenzeller 88% (n02107908-Appenzeller)

n02097130-giant_schnauzer 71% (n02097130-giant_schnauzer)

Hyperparameter Tuning

Model	Accuracy
Model 1: ● ImageNet ● Global Average Layer ● Dropout .2 ● Dense 120 Softmax	Train: 0.7414 Test: 0.8350
Model 2: ● ImageNet ● Global Average Layer ● Dropout .2 ● Dense 1000 RELU ● Dropout .2 ● Dense 120 Softmax	Train: 0.7735 Test: 0.8292
Model 3: ● ImageNet ● Global Average Layer ● Dropout .2 ● Dense 1000 Sigmoid ● Dropout .2 ● Dense 120 Softmax	Train: 0.7359 Test: 0.8348
Model 4: ● ImageNet ● Global Average Layer ● Dropout .2 ● Dense 1000 RELU ● Dropout .2 ● Dense 500 RELU ● Dropout .2 ● Dense 120 Softmax	Train: 0.7397 Test: 0.8275
Model 5 ● ImageNet ● Global Average Layer ● Dropout .2 ● Dense 1000 Sigmoid ● Dropout .2 ● Dense 500 RELU ● Dropout .2 ● Dense 120 Softmax	Train: .7059 Test: .8161

Model 6 ● ImageNet ● GlobalAverageLayer ● Dropout .2 ● Dense 1000 Sigmoid ● Dropout .2 ● Dense 1000 Sigmoid ● Dropout .2 ● Dense 120 Softmax	Train: .6856 Test: .8088