SEP 789 – Deep Learning and Its Applications

Assignment 1

Submitted by,

Greeshma Gopal(gopalg)
ID- 400245291

Dog and cat example

- In the example given, the dataset is a set of images of cats and dogs. The aim is to make the machine learn the difference between cats and dogs and later identify the same.

- The jupyter notebook settings would be changed to TPU for the code to run.

- The dataset has been uploaded in the google drive and is being accessed. Since the dataset has already been uploaded already, I did not run the code to unzipping(it would be showing error in the jupyter code attached with the assignment).

- The data has been split into train, test and validation batches. Training data – 8005 images, Validation data- 2023 images and Testing data- 20 images.

- The convolution layer was defined with the number of features, shape of the image and the gray scale image of 224*224*1.

- Activation function used is Relu in all the dense layers. And when compiling the optimizer used is Adam.

- By default, the jupyter file had 3 epochs, However I wanted to test with 5 epochs. Below are the results of the same. The accuracy was decreasing until 3rd epoch and starting increasing again, which could be because the model was getting overfitted. Hence, the ideal number of epochs should be 3.

- The training accuracy, at the 5th epoch is 68% and for the accuracy for validation dataset is 62%. Here we are calculating categorical accuracy because we have two categories dogs and cats. Both softmax and sigmoid were used as activation function. It was found that the accuracies were more for softmax.

Softmax

```
19]
    Epoch 1/5
    WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:190: The

    WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:207: The

    WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:216: The

    WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:223: The

     - 112s - loss: 0.7071 - categorical_accuracy: 0.5660 - val_loss: 0.5218 - val_categorical_accuracy: 0.8750
    Epoch 2/5
     - 110s - loss: 0.6259 - categorical_accuracy: 0.6507 - val_loss: 0.5527 - val_categorical_accuracy: 0.7500
    Epoch 3/5
     - 111s - loss: 0.6033 - categorical_accuracy: 0.6780 - val_loss: 0.4897 - val_categorical_accuracy: 0.6875
    Epoch 4/5
     - 110s - loss: 0.5903 - categorical_accuracy: 0.6787 - val_loss: 0.7483 - val_categorical_accuracy: 0.5000
    Epoch 5/5
     - 108s - loss: 0.5967 - categorical_accuracy: 0.6827 - val_loss: 0.5311 - val_categorical_accuracy: 0.6250
    <keras.callbacks.History at 0x7f21703a02b0>
```

Sigmoid

```
Epoch 1/5
 - 113s - loss: 7.9562 - categorical_accuracy: 0.5020 - val_loss: 11.0812 - val_categorical_accuracy: 0.3125
Epoch 2/5
 - 112s - loss: 7.9838 - categorical_accuracy: 0.5047 - val_loss: 11.0812 - val_categorical_accuracy: 0.3125
Epoch 3/5
 - 111s - loss: 7.9945 - categorical_accuracy: 0.5040 - val_loss: 8.0590 - val_categorical_accuracy: 0.5000
Epoch 4/5
 - 112s - loss: 8.2847 - categorical_accuracy: 0.4860 - val_loss: 8.0590 - val_categorical_accuracy: 0.5000
Epoch 5/5
 - 114s - loss: 8.0053 - categorical_accuracy: 0.5033 - val_loss: 9.0664 - val_categorical_accuracy: 0.4375
<keras.callbacks.History at 0x7f1847937a20>
```

- The model was then tested and saved.

```
[ ] predictor[:,0]
```

```
array([0.64021325, 0.53214055, 0.72681135, 0.11416289], dtype=float32)
```

```
[ ] from sklearn.metrics import accuracy_score
    accuracy_score(test_label[:,0], predictor[:,0].round(), normalize=False)
```

```
3
```

```
[ ] confusion_matrix(test_label[:,0],predictor[:,0].round())
```

```
array([[1, 1],
       [0, 2]])
```

```
[ ] model.save('dog_cat_CNN.model')
```

Transfer Learning observations

- The dataset has been already loaded in the google drive

- While dividing the dataset into training, test and validation, I have changed the color_mode to ‘rgb’ instead of ‘grayscale’ since it is 3 dimensional data.

- VGG16 model was created. The optimizer used is Adam and number of epochs is 10. The accuracy we got is 60% for the training data. VGG can be used as a CNN architecture to get better accuracy and less computational times.

```
Epoch 1/10
 - 107s - loss: 5.4367 - acc: 0.4750 - val_loss: 7.2531 - val_acc: 0.5500
Epoch 2/10
 - 101s - loss: 9.2679 - acc: 0.4250 - val_loss: 8.8650 - val_acc: 0.4500
Epoch 3/10
 - 101s - loss: 7.6561 - acc: 0.5250 - val_loss: 9.2679 - val_acc: 0.4250
Epoch 4/10
 - 101s - loss: 6.8502 - acc: 0.5750 - val_loss: 8.8650 - val_acc: 0.4500
Epoch 5/10
 - 101s - loss: 9.6709 - acc: 0.4000 - val_loss: 8.0590 - val_acc: 0.5000
Epoch 6/10
 - 101s - loss: 6.8502 - acc: 0.5750 - val_loss: 9.2679 - val_acc: 0.4250
Epoch 7/10
 - 101s - loss: 6.4472 - acc: 0.6000 - val_loss: 7.2531 - val_acc: 0.5500
Epoch 8/10
 - 101s - loss: 8.8650 - acc: 0.4500 - val_loss: 9.2679 - val_acc: 0.4250
Epoch 9/10
 - 101s - loss: 8.0590 - acc: 0.5000 - val_loss: 5.2384 - val_acc: 0.6750
Epoch 10/10
 - 103s - loss: 7.6561 - acc: 0.5250 - val_loss: 6.4472 - val_acc: 0.6000
<keras.callbacks.History at 0x7facf4f87eb8>
```

- In the second section, we are building VGG16 model with freeze layers. Here drop out is used and optimizer is rmsprop. When comparing the accuracies of VGG16 model with freeze and without freeze, we can find that the accuracies are less for the model with freezing. This is because, the dataset is very much different from the imagenet dataset.

- When VGG16 is run with sequential model, we can see the accuracies are better than when the models are run with VGG16 freeze and VGG16 initial weight. However, the computational times with sequential is higher.

- For VGG16 freeze, the accuracies are comparatively lesser because we are not updating the weights. Also the computational time is very less since these weights are not being updated.