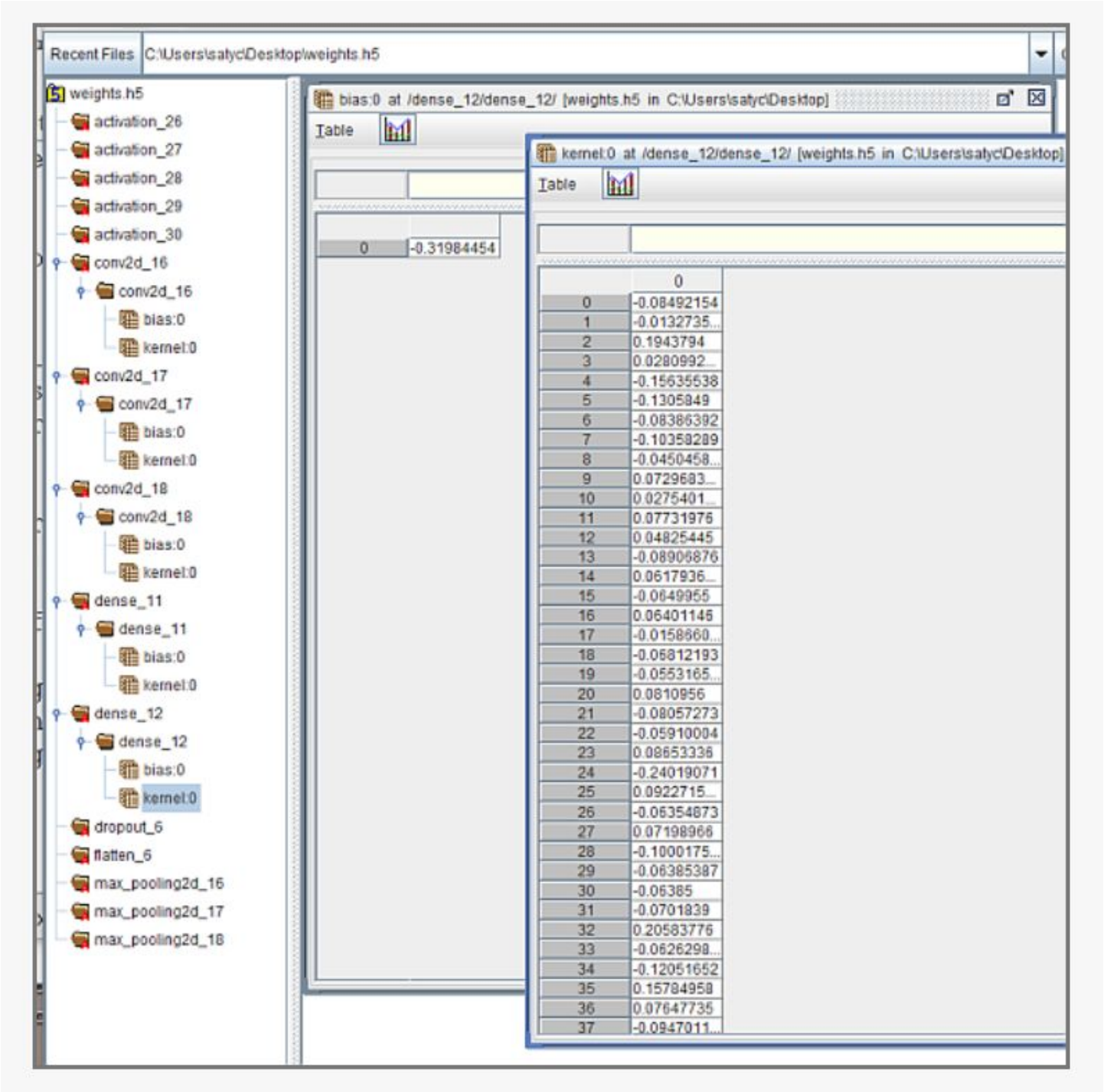
HW 5 Rubrics

Total: 5 points
Late Penalty: 0.5/day, after 29th April 2020 11:59 pm
Files have not been zipped penalty: 0.5

Q1: 1 point

0.5 pt for submitting weights.h5 file

0.5 pt for submitting a screengrab of any dense layer



The screengrabs should show the expanded results of any dense layer below weights.h5

-0.5 points if weights.h5 is not submitted

-0.5 pt if the screenshot is not submitted

Q2: 2 points

1 pt for a screengrab showing correct classification. The values with a 20% range are accepted. So, 0.8 and 0.2 are fine.

0.5 pt each for good_cat and good_dog images

classify.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
from keras.preprocessing import image



myPic = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/good_cat.jpg'
test_image= image.load_img(myPic, target_size = (img_width, img_height))
test_image = image.img_to_array(test_image)
test_image = test_image.reshape(input_shape)
test_image = numpy.expand_dims(test_image, axis = 0)
result = model.predict(test_image,verbose=0)
print(result[0][0])

myPic2 = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/good_dog.jpg'
#myPic2 = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/corgi.jpg'
test_image2= image.load_img(myPic2, target_size = (img_width, img_height))
test_image2 = image.img_to_array(test_image2)
test_image2 = test_image2.reshape(input_shape)
test_image2 = numpy.expand_dims(test_image2, axis = 0)
result = model.predict(test_image2,verbose=0)
print(result[0][0])
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
0.0
1.0
```

[ ]

-1 pt if the classification is wrong or screenshot is not submitted

-0.5 pt each for wrong dog and cat images

Q3: 2 points

1 pt for a screengrab showing misclassification. The values with a 20% range are accepted. So, 0.8 and 0.2 are fine.

0.5 pt each for trick_cat and trick_dog images

classify.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

```
                metrics=['accuracy'])

#from keras.models import load_model
model.load_weights('/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/weights.h5')



from keras.preprocessing import image



myPic = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/trick_cat.jpg'
test_image= image.load_img(myPic, target_size = (img_width, img_height))
test_image = image.img_to_array(test_image)
test_image = test_image.reshape(input_shape)
test_image = numpy.expand_dims(test_image, axis = 0)
result = model.predict(test_image,verbose=0)
print(result[0][0])

myPic2 = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/trick_dog.jpg'
#myPic2 = '/content/drive/My Drive/Colab Notebooks/cats-vs-dogs/data/live/corgi.jpg'
test_image2= image.load_img(myPic2, target_size = (img_width, img_height))
test_image2 = image.img_to_array(test_image2)
test_image2 = test_image2.reshape(input_shape)
test_image2 = numpy.expand_dims(test_image2, axis = 0)
result = model.predict(test_image2,verbose=0)
print(result[0][0])
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
1.0
0.0
```

-1 pt if the classification is wrong or screenshot is not submitted

-0.5 pt each for wrong cat and dog images

Bonus Question: No points

The dataset should contain at least 300 images

Screengrab of correct classification

Readme.txt file containing the changes made to the code

Readme file should contain the following changes:

1) model.add(Dense(3))

Indicates the dimensionality of the output space. Our output space must be of the form [0, 0, 1] In other words, they can be considered as the output nodes for each possible class.

2) model.add(Activation('softmax')) :

It is the last layer activation function, it returns an array of 3 probability scores summing to 1. The sigmoid function is used for the two-class classification, whereas the softmax function is used for the multiclass classification.

3) model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])

Since we are dealing with multiple classes (non-binary), we change the loss function from binary_crossentropy to categorical_crossentropy

4) train_generator = train_datagen.flow_from_directory(train_data_dir, target_size=(img_width, img_height), batch_size=batch_size, class_mode='categorical')

Since we are trying to train the neural network with multiple classes (non-binary), we are using the categorical class mode instead of the binary class mode.

5) validation_generator = test_datagen.flow_from_directory(

validation_data_dir,

target_size=(img_width, img_height),

batch_size=batch_size,

class_mode='categorical')

Since we are trying to test the neural network with multiple classes (non-binary), we are using the categorical class mode instead of binary class mode.