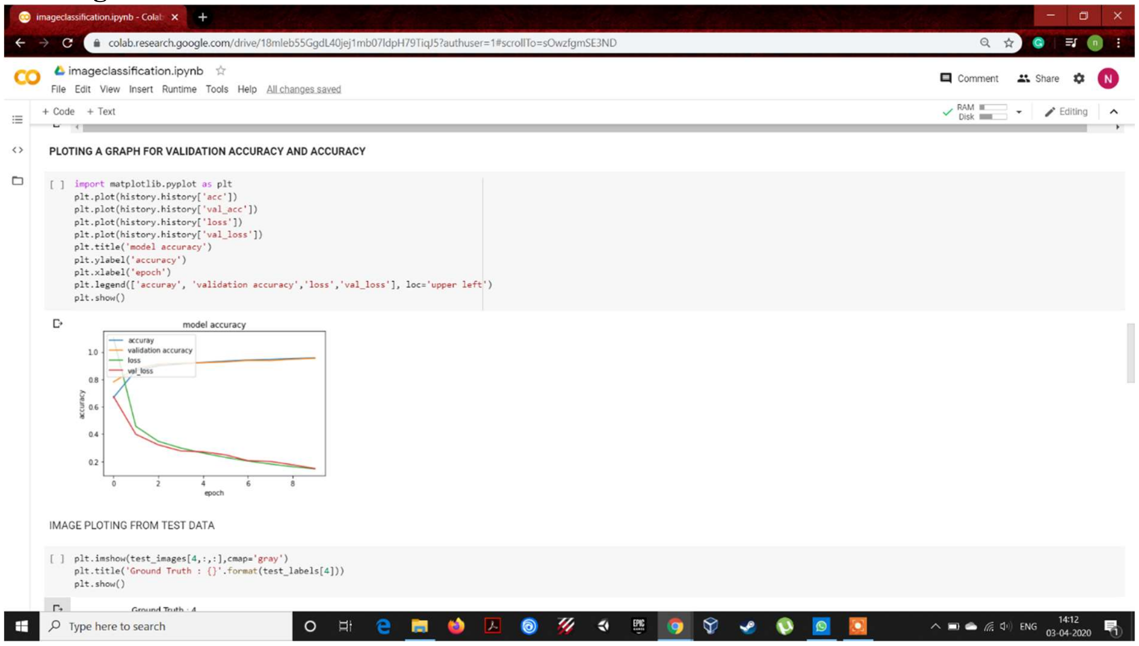
MICP2

1. **using the history object in the source code, plot the loss and accuracy for both training data and validation data.**


imageclassification.ipynb - Colab
colab.research.google.com/drive/18mleb55GgdL40jej1mb07ldpH79TiqJ5?authuser=1#scrollTo=sOwzfgmSE3ND
imageclassification.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
model accuracy
accuray
validation accuracy
loss
val_loss
accuracy
epoch
Type here to search
ENG
14:12
03-04-2020

PLOTING A GRAPH FOR VALIDATION ACCURACY AND ACCURACY

```
import matplotlib.pyplot as plt
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['accuray', 'validation accuracy','loss','val_loss'], loc='upper left')
plt.show()
```

IMAGE PLOTING FROM TEST DATA

```
plt.imshow(test_images[4,:,:],cmap='gray')
plt.title('Ground Truth : {}'.format(test_labels[4]))
plt.show()
```

2. **plot one of the images in the test data, and then do inferencing to check what is the prediction of the model on that single image in the test data**

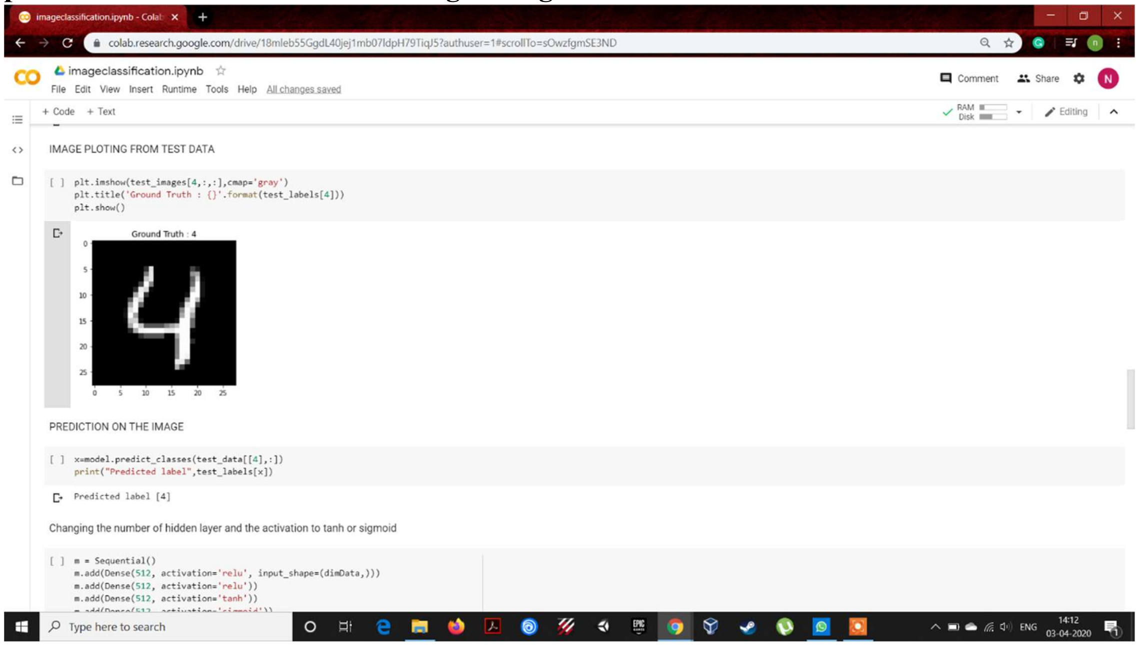

imageclassification.ipynb - Colab
colab.research.google.com/drive/18mleb55GgdL40jej1mb07ldpH79TiqJ5?authuser=1#scrollTo=sOwzfgmSE3ND
imageclassification.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
Ground Truth : 4
Type here to search
ENG
14:12
03-04-2020

IMAGE PLOTING FROM TEST DATA

```
plt.imshow(test_images[4,:,:],cmap='gray')
plt.title('Ground Truth : {}'.format(test_labels[4]))
plt.show()
```

PREDICTION ON THE IMAGE

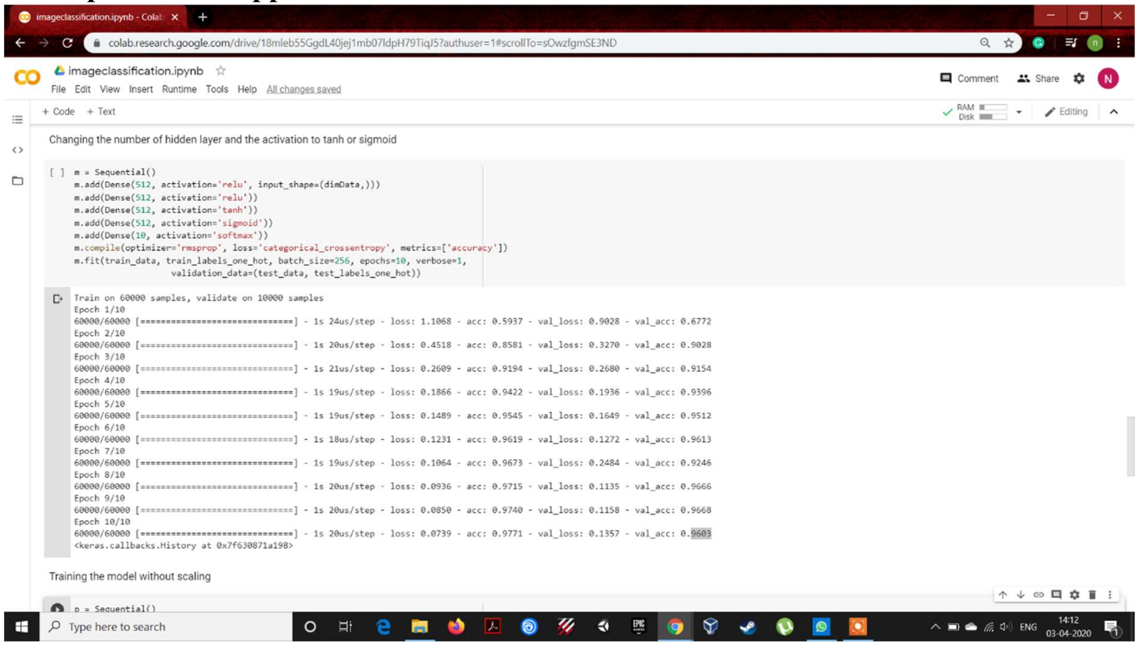
```
x=model.predict_classes(test_data[[4],:])
print("Predicted label",test_labels[x])
```

```
Predicted label [4]
```

Changing the number of hidden layer and the activation to tanh or sigmoid

```
m = Sequential()
m.add(Dense(512, activation='relu', input_shape=(dimData,)))
m.add(Dense(512, activation='relu'))
m.add(Dense(512, activation='tanh'))
```

3. We had used 2 hidden layers and relu activation:
a. Try to change the number of hidden layer and the activation to tanh or sigmoid and report what happens.


imageclassification.ipynb - Colab
colab.research.google.com/drive/18mleb55GgdL40jej1mb07ldpH79TiqJ5?authuser=1#scrollTo=sOwzfgmSE3ND
imageclassification.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Comment Share
Editing
Type here to search
14:12
03-04-2020

Changing the number of hidden layer and the activation to tanh or sigmoid

```
m = Sequential()
m.add(Dense(512, activation='relu', input_shape=(dimData,)))
m.add(Dense(512, activation='relu'))
m.add(Dense(512, activation='tanh'))
m.add(Dense(512, activation='sigmoid'))
m.add(Dense(10, activation='softmax'))
m.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
m.fit(train_data, train_labels_one_hot, batch_size=256, epochs=10, verbose=1,
                   validation_data=(test_data, test_labels_one_hot))
```

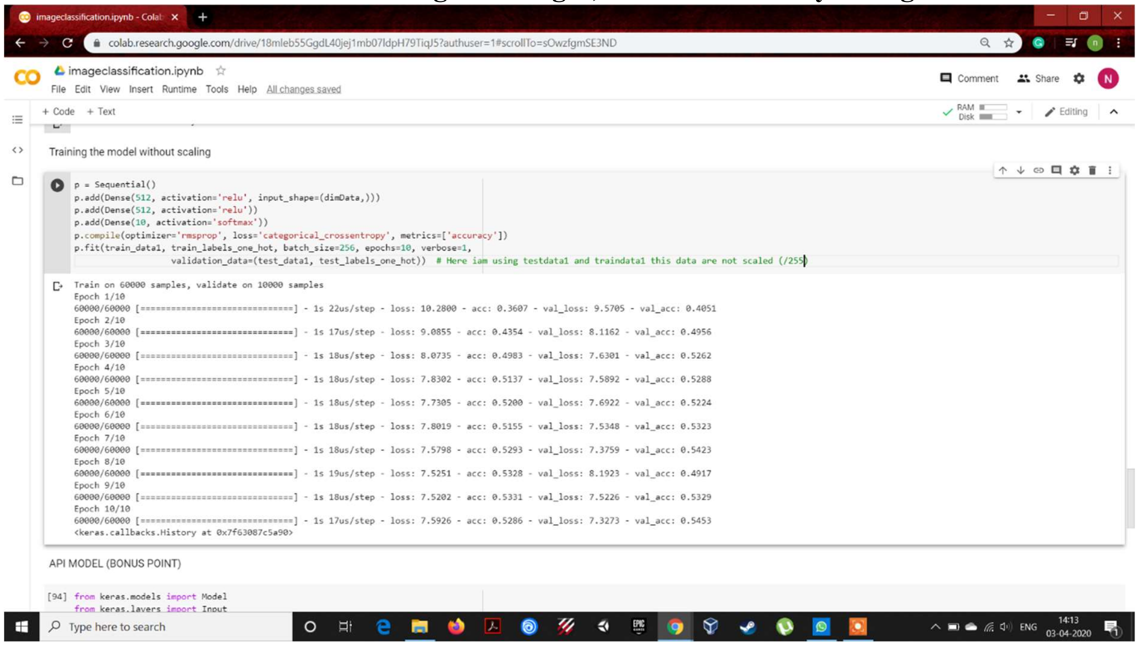
```
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 1s 24us/step - loss: 1.1068 - acc: 0.5937 - val_loss: 0.9028 - val_acc: 0.6772
Epoch 2/10
60000/60000 [==============================] - 1s 20us/step - loss: 0.4518 - acc: 0.8581 - val_loss: 0.3270 - val_acc: 0.9028
Epoch 3/10
60000/60000 [==============================] - 1s 21us/step - loss: 0.2609 - acc: 0.9194 - val_loss: 0.2680 - val_acc: 0.9154
Epoch 4/10
60000/60000 [==============================] - 1s 19us/step - loss: 0.1866 - acc: 0.9422 - val_loss: 0.1936 - val_acc: 0.9396
Epoch 5/10
60000/60000 [==============================] - 1s 19us/step - loss: 0.1489 - acc: 0.9545 - val_loss: 0.1649 - val_acc: 0.9512
Epoch 6/10
60000/60000 [==============================] - 1s 18us/step - loss: 0.1231 - acc: 0.9619 - val_loss: 0.1272 - val_acc: 0.9613
Epoch 7/10
60000/60000 [==============================] - 1s 19us/step - loss: 0.1064 - acc: 0.9673 - val_loss: 0.2484 - val_acc: 0.9246
Epoch 8/10
60000/60000 [==============================] - 1s 20us/step - loss: 0.0936 - acc: 0.9715 - val_loss: 0.1135 - val_acc: 0.9666
Epoch 9/10
60000/60000 [==============================] - 1s 20us/step - loss: 0.0850 - acc: 0.9740 - val_loss: 0.1158 - val_acc: 0.9668
Epoch 10/10
60000/60000 [==============================] - 1s 20us/step - loss: 0.0739 - acc: 0.9771 - val_loss: 0.1357 - val_acc: 0.9603
<keras.callbacks.History at 0x7f630871a198>
```

Training the model without scaling

```
p = Sequential()
```

After adding more hidden layers the accuracy increased

4. Run the same code without scaling the images, how the accuracy changes?


imageclassification.ipynb - Colab
colab.research.google.com/drive/18mleb55GgdL40jej1mb07ldpH79TiqJ5?authuser=1#scrollTo=sOwzfgmSE3ND
imageclassification.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Comment Share
Editing
Type here to search
14:13
03-04-2020

Training the model without scaling

```
p = Sequential()
p.add(Dense(512, activation='relu', input_shape=(dimData,)))
p.add(Dense(512, activation='relu'))
p.add(Dense(10, activation='softmax'))
p.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
p.fit(train_data1, train_labels_one_hot, batch_size=256, epochs=10, verbose=1,
                   validation_data=(test_data1, test_labels_one_hot))  # Here iam using testdata1 and traindata1 this data are not scaled (/255)
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 1s 22us/step - loss: 10.2800 - acc: 0.3607 - val_loss: 9.5705 - val_acc: 0.4051
Epoch 2/10
60000/60000 [==============================] - 1s 17us/step - loss: 9.0855 - acc: 0.4354 - val_loss: 8.1162 - val_acc: 0.4956
Epoch 3/10
60000/60000 [==============================] - 1s 18us/step - loss: 8.0735 - acc: 0.4983 - val_loss: 7.6301 - val_acc: 0.5262
Epoch 4/10
60000/60000 [==============================] - 1s 18us/step - loss: 7.8302 - acc: 0.5137 - val_loss: 7.5892 - val_acc: 0.5288
Epoch 5/10
60000/60000 [==============================] - 1s 18us/step - loss: 7.7305 - acc: 0.5200 - val_loss: 7.6922 - val_acc: 0.5224
Epoch 6/10
60000/60000 [==============================] - 1s 18us/step - loss: 7.8019 - acc: 0.5155 - val_loss: 7.5348 - val_acc: 0.5323
Epoch 7/10
60000/60000 [==============================] - 1s 18us/step - loss: 7.5798 - acc: 0.5293 - val_loss: 7.3759 - val_acc: 0.5423
Epoch 8/10
60000/60000 [==============================] - 1s 19us/step - loss: 7.5251 - acc: 0.5328 - val_loss: 8.1923 - val_acc: 0.4917
Epoch 9/10
60000/60000 [==============================] - 1s 18us/step - loss: 7.5202 - acc: 0.5331 - val_loss: 7.5226 - val_acc: 0.5329
Epoch 10/10
60000/60000 [==============================] - 1s 17us/step - loss: 7.5926 - acc: 0.5286 - val_loss: 7.3273 - val_acc: 0.5453
<keras.callbacks.History at 0x7f63087c5a90>
```

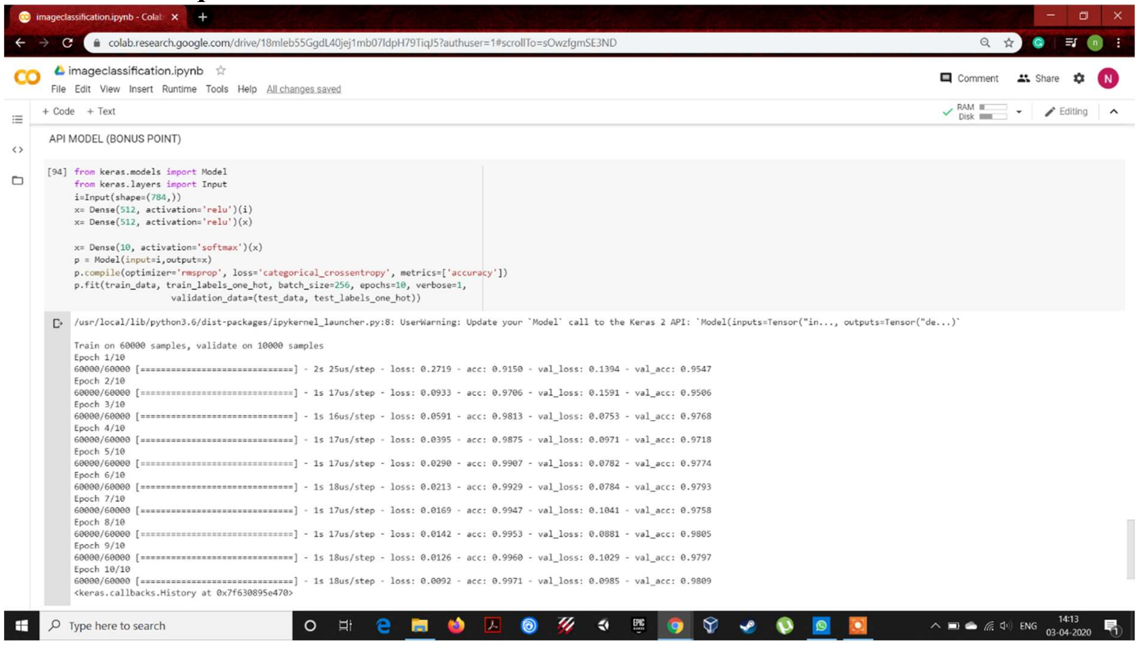
API MODEL (BONUS POINT)

```
[94] from keras.models import Model
     from keras.layers import Input
```

After training with unscaled data, the accuracy got declined from 95% to 52%

<u> Bonus point:</u>**

5. **Convert the sequential model to API model.**


imageclassification.ipynb - Colab
colab.research.google.com/drive/18mleb55GgdL40jej1mb07ldpH79TiqJ5?authuser=1#scrollTo=sOwzfgmSE3ND
imageclassification.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Editing
Type here to search
ENG
14:13
03-04-2020

API MODEL (BONUS POINT)

```
[94] from keras.models import Model
     from keras.layers import Input
     i=Input(shape=(784,))
     x= Dense(512, activation='relu')(i)
     x= Dense(512, activation='relu')(x)

     x= Dense(10, activation='softmax')(x)
     p = Model(input=i,output=x)
     p.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
     p.fit(train_data, train_labels_one_hot, batch_size=256, epochs=10, verbose=1,
                       validation_data=(test_data, test_labels_one_hot))
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: UserWarning: Update your `Model` call to the Keras 2 API: `Model(inputs=Tensor("in..., outputs=Tensor("de...)`

Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 2s 25us/step - loss: 0.2719 - acc: 0.9150 - val_loss: 0.1394 - val_acc: 0.9547
Epoch 2/10
60000/60000 [==============================] - 1s 17us/step - loss: 0.0933 - acc: 0.9706 - val_loss: 0.1591 - val_acc: 0.9506
Epoch 3/10
60000/60000 [==============================] - 1s 16us/step - loss: 0.0591 - acc: 0.9813 - val_loss: 0.0753 - val_acc: 0.9768
Epoch 4/10
60000/60000 [==============================] - 1s 17us/step - loss: 0.0395 - acc: 0.9875 - val_loss: 0.0971 - val_acc: 0.9718
Epoch 5/10
60000/60000 [==============================] - 1s 17us/step - loss: 0.0290 - acc: 0.9907 - val_loss: 0.0782 - val_acc: 0.9774
Epoch 6/10
60000/60000 [==============================] - 1s 18us/step - loss: 0.0213 - acc: 0.9929 - val_loss: 0.0784 - val_acc: 0.9793
Epoch 7/10
60000/60000 [==============================] - 1s 17us/step - loss: 0.0169 - acc: 0.9947 - val_loss: 0.1041 - val_acc: 0.9758
Epoch 8/10
60000/60000 [==============================] - 1s 17us/step - loss: 0.0142 - acc: 0.9953 - val_loss: 0.0881 - val_acc: 0.9805
Epoch 9/10
60000/60000 [==============================] - 1s 18us/step - loss: 0.0126 - acc: 0.9960 - val_loss: 0.1029 - val_acc: 0.9797
Epoch 10/10
60000/60000 [==============================] - 1s 18us/step - loss: 0.0092 - acc: 0.9971 - val_loss: 0.0985 - val_acc: 0.9809
<keras.callbacks.History at 0x7f630895e470>
```

By
DUKKIPATI SRI SAI NITHIN CHOWDARY
Class id: 4
Team id: 6