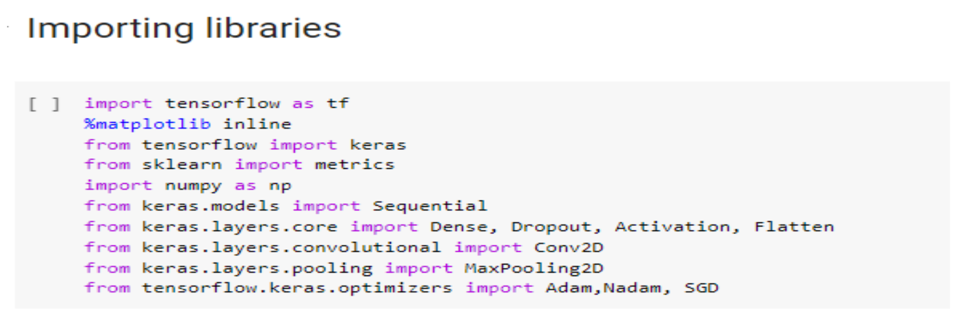
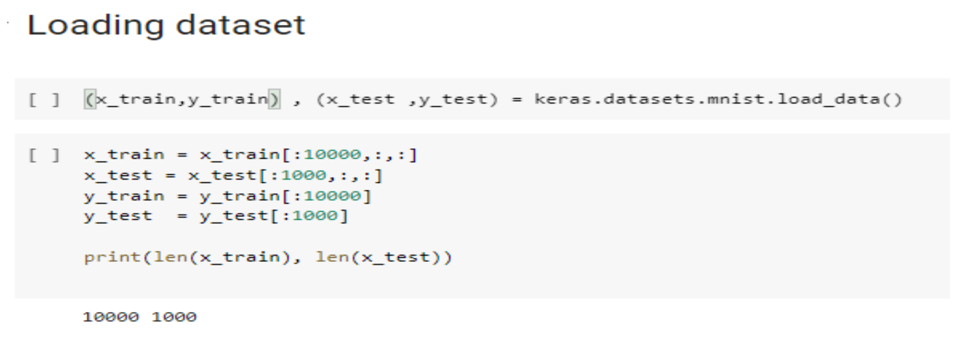
Abanoub Samir Girgis || 20190001
Joyce Fayek Milad || 20190160
Assignment 3
Group: S2



Importing libraries

```
[ ] import tensorflow as tf
    %matplotlib inline
    from tensorflow import keras
    from sklearn import metrics
    import numpy as np
    from keras.models import Sequential
    from keras.layers.core import Dense, Dropout, Activation, Flatten
    from keras.layers.convolutional import Conv2D
    from keras.layers.pooling import MaxPooling2D
    from tensorflow.keras.optimizers import Adam,Nadam, SGD
```

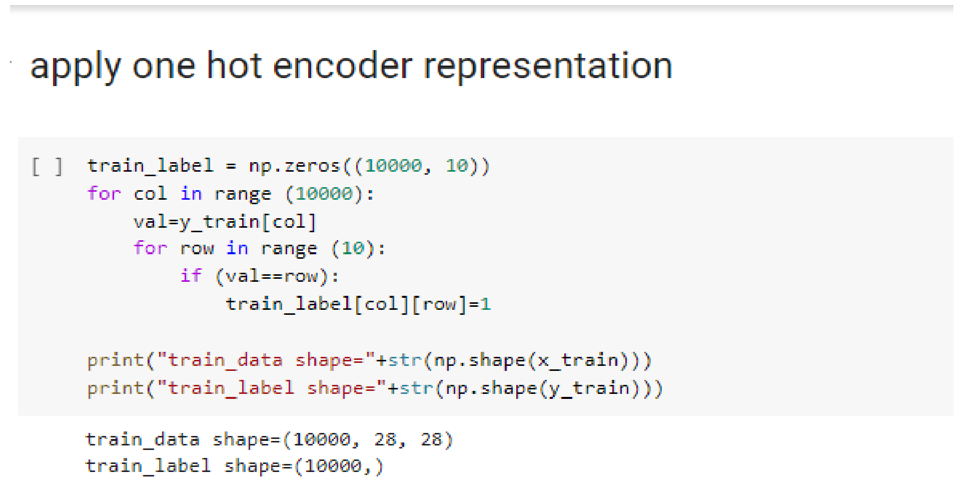


Loading dataset

```
[ ] (x_train,y_train) , (x_test ,y_test) = keras.datasets.mnist.load_data()
```

```
[ ] x_train = x_train[:10000,:,:]
    x_test = x_test[:1000,:,:]
    y_train = y_train[:10000]
    y_test  = y_test[:1000]

    print(len(x_train), len(x_test))
```

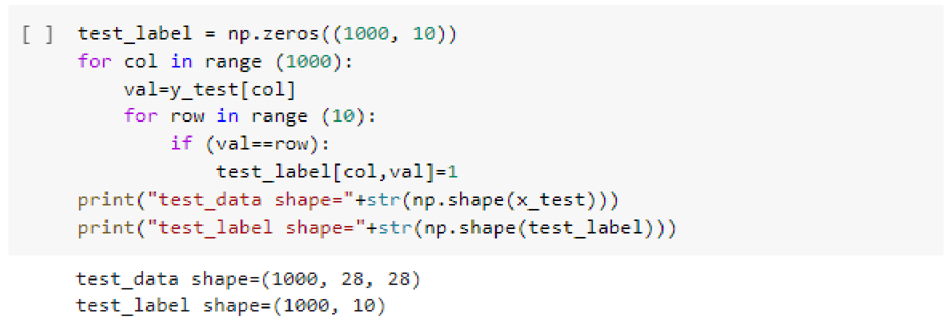
```
10000 1000
```



apply one hot encoder representation

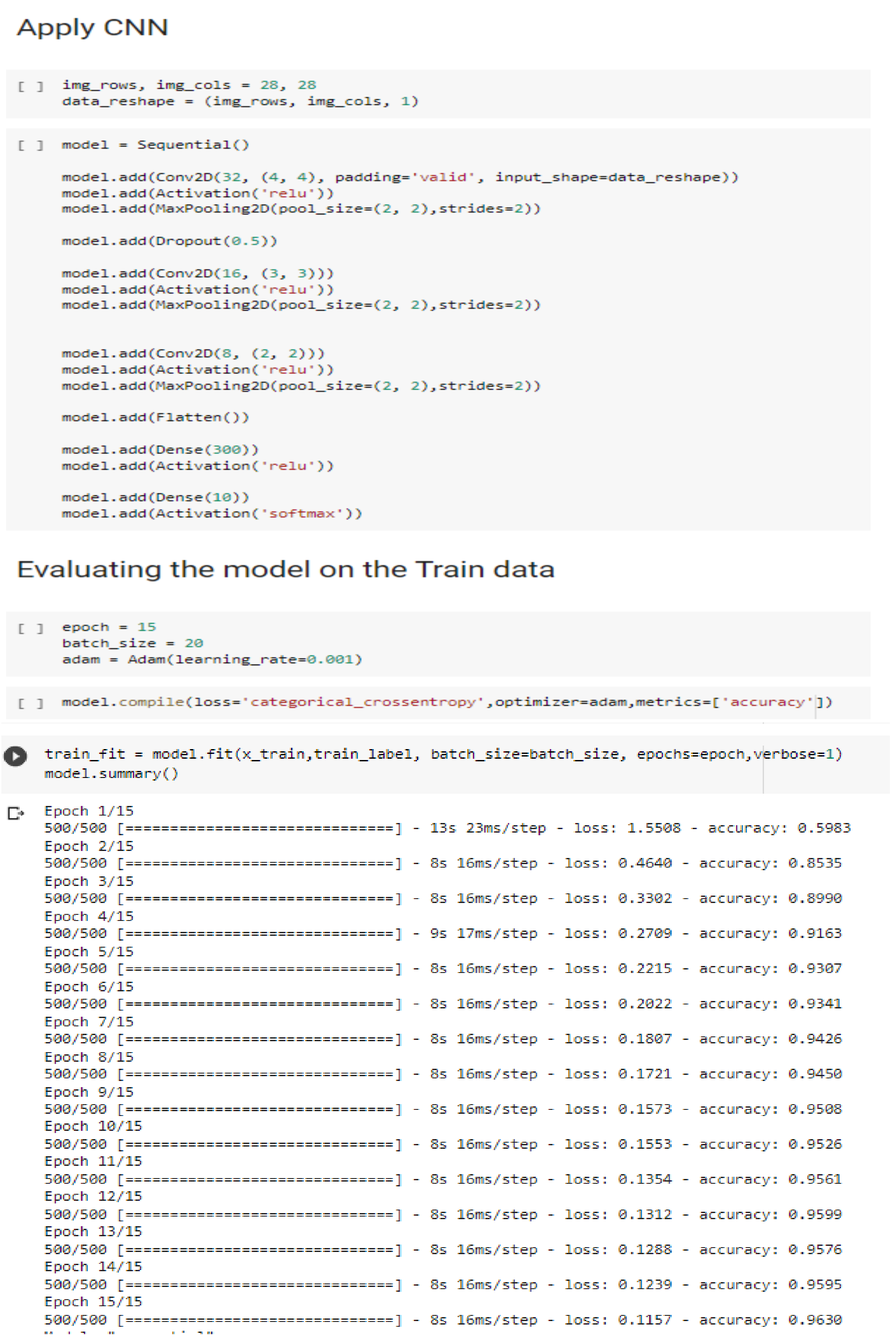
```
[ ] train_label = np.zeros((10000, 10))
    for col in range (10000):
        val=y_train[col]
        for row in range (10):
            if (val==row):
                train_label[col][row]=1

    print("train_data shape="+str(np.shape(x_train)))
    print("train_label shape="+str(np.shape(y_train)))
```

```
train_data shape=(10000, 28, 28)
train_label shape=(10000,)
```



```
[ ] test_label = np.zeros((1000, 10))
    for col in range (1000):
        val=y_test[col]
        for row in range (10):
            if (val==row):
                test_label[col,val]=1
    print("test_data shape="+str(np.shape(x_test)))
    print("test_label shape="+str(np.shape(test_label)))
```

```
test_data shape=(1000, 28, 28)
test_label shape=(1000, 10)
```



Apply CNN

```
[ ] img_rows, img_cols = 28, 28
    data_reshape = (img_rows, img_cols, 1)
```

```
[ ] model = Sequential()

    model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2),strides=2))

    model.add(Dropout(0.5))

    model.add(Conv2D(16, (3, 3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2),strides=2))


    model.add(Conv2D(8, (2, 2)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2),strides=2))

    model.add(Flatten())

    model.add(Dense(300))
    model.add(Activation('relu'))

    model.add(Dense(10))
    model.add(Activation('softmax'))
```

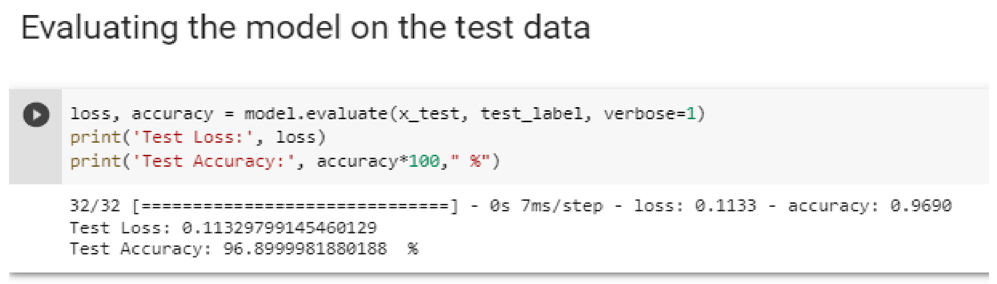
Evaluating the model on the Train data

```
[ ] epoch = 15
    batch_size = 20
    adam = Adam(learning_rate=0.001)
```

```
[ ] model.compile(loss='categorical_crossentropy',optimizer=adam,metrics=['accuracy'])
```

```
train_fit = model.fit(x_train,train_label, batch_size=batch_size, epochs=epoch,verbose=1)
model.summary()
```

```
Epoch 1/15
500/500 [==============================] - 13s 23ms/step - loss: 1.5508 - accuracy: 0.5983
Epoch 2/15
500/500 [==============================] - 8s 16ms/step - loss: 0.4640 - accuracy: 0.8535
Epoch 3/15
500/500 [==============================] - 8s 16ms/step - loss: 0.3302 - accuracy: 0.8990
Epoch 4/15
500/500 [==============================] - 9s 17ms/step - loss: 0.2709 - accuracy: 0.9163
Epoch 5/15
500/500 [==============================] - 8s 16ms/step - loss: 0.2215 - accuracy: 0.9307
Epoch 6/15
500/500 [==============================] - 8s 16ms/step - loss: 0.2022 - accuracy: 0.9341
Epoch 7/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1807 - accuracy: 0.9426
Epoch 8/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1721 - accuracy: 0.9450
Epoch 9/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1573 - accuracy: 0.9508
Epoch 10/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1553 - accuracy: 0.9526
Epoch 11/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1354 - accuracy: 0.9561
Epoch 12/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1312 - accuracy: 0.9599
Epoch 13/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1288 - accuracy: 0.9576
Epoch 14/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1239 - accuracy: 0.9595
Epoch 15/15
500/500 [==============================] - 8s 16ms/step - loss: 0.1157 - accuracy: 0.9630
```



Evaluating the model on the test data

```
loss, accuracy = model.evaluate(x_test, test_label, verbose=1)
print('Test Loss:', loss)
print('Test Accuracy:', accuracy*100," %")
```

```
32/32 [==============================] - 0s 7ms/step - loss: 0.1133 - accuracy: 0.9690
Test Loss: 0.11329799145460129
Test Accuracy: 96.8999981880188  %
```

The Effect of using different Number of Epochs:

Model 1:

- Epoch = 10
- Batch Size = 32

Testing:

1. Final Accuracy: 96.399%
   For train:
   - Epoch 1: accuracy: 0.4529
   - Epoch 2: accuracy: 0.8976
   - Epoch 3: accuracy: 0.9369
   - Epoch 4: accuracy: 0.9613
   - Epoch 5: accuracy: 0.9700

   For test:
   - Accuracy: 0.96021

2. Conv1 = $(5\ X\ 5\ X\ 1)64 + 64 = 1664$
   Conv2 = $(4\ X\ 4\ X\ 64)\ 32 + 32 = 32800$
   FC_hidden1= $(4X4X32)\ X\ 512 + 512 = \ 262656$
   FC_hidden2 = $512\ X\ 206 + 206 = 105678$
   FC_output = $206\ X\ 10 + 10 = 2070$
   Number of Parameters = 404868 parameters

3. Average time for Train:
   Epoch 1: 25s 73ms/step
   Epoch 2: 19s 60ms/ step
   Epoch 3: 19s 61ms/step
   Epoch 4: 18s 59ms/step
   Epoch 5: 19s 60ms/step
   Epoch 6: 20s 63ms/step
   Epoch 7: 19s 61ms/step
   Epoch 8: 19s 60ms/step
   Epoch 9: 19s 60ms/step
   Epoch 10: 19s 60ms/step

4. Average time for Test = 1s 16ms/step

5. Conv1: Filters=64, kernel Size= 5 x 5, activation function= “Relu”
   Conv2: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   FC_hidden1: Size=512, activation function= “Relu”
   FC_hidden2: Size=206, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

Model 2:

- Epoch = 13
- Batch Size = 32

Testing:

1. Final Accuracy: 96.899%
   For train:
   - Epoch 1: accuracy: 0.9291
   - Epoch 2: accuracy: 0.9506
   - Epoch 3: accuracy: 0.9706
   - Epoch 4: accuracy: 0.9729
   - Epoch 5: accuracy: 0.9791

   For test:

   - Accuracy: 0.9690

2. Average time for Train:
   Epoch 1: 21s 64ms/step
   Epoch 2: 20s 63ms/step
   Epoch 3: 19s 60ms/step
   Epoch 4: 19s 62ms/step
   Epoch 5: 19s 61ms/step
   Epoch 6: 19s 61ms/step
   Epoch 7: 19s 61ms/step
   Epoch 8: 19s 62ms/step

Epoch 9: 19s 62ms/step
Epoch 10: 19s 61ms/step
Epoch 11: 19s 61ms/step
Epoch 12: 19s 61ms/step
Epoch 13: 19s 64ms/step

3. Average time for Test = 1s 20ms/step

6. Learning Rate=0.001
Optimizer: Momentum=0.9

Model 3:

- Epoch = 15
- Batch Size = 32

1. Final Accuracy: 98.5%
   For train:
   - Epoch 1: accuracy: 0.9971
   - Epoch 2: accuracy: 0.9985
   - Epoch 3: accuracy: 0.9997
   - Epoch 4: accuracy: 1.000
   - Epoch 5: accuracy: 1.000

   For test:

   8. Accuracy: 0.9850

3. Average time for Train:
   Epoch 1: 21s 65ms/step
   Epoch 2: 20s 63ms/step
   Epoch 3: 19s 61ms/step
   Epoch 4: 20s 62ms/step
   Epoch 5: 19s 61ms/step
   Epoch 6: 19s 63ms/step
   Epoch 7: 19s 61ms/step
   Epoch 8: 19s 67ms/step
   Epoch 9: 20s 62ms/step
   Epoch 10: 19s 64ms/step
   Epoch 11: 19s 61ms/step

Epoch 12: 19s 63ms/step
Epoch 13: 19s 64ms/step
Epoch 14: 20s 64ms/step
Epoch 15: 19s 60ms/step

4. Average time for Test = 1s 24ms/step
6. Learning Rate=0.001
   Optimizer: Momentum=0.9

Conclusion:
When using epochs = 10 we got accuracy = 96.399% (model 1)
and when using epochs = 13 the accuracy was 96.89 % (model 2)
and when using epochs = 15 the accuracy was 98.5 % (model 3)
the higher the epochs, the higher the accuracy
So, the best case was with Epoch = 15

The Effect of using different learning rates:

Model 4:

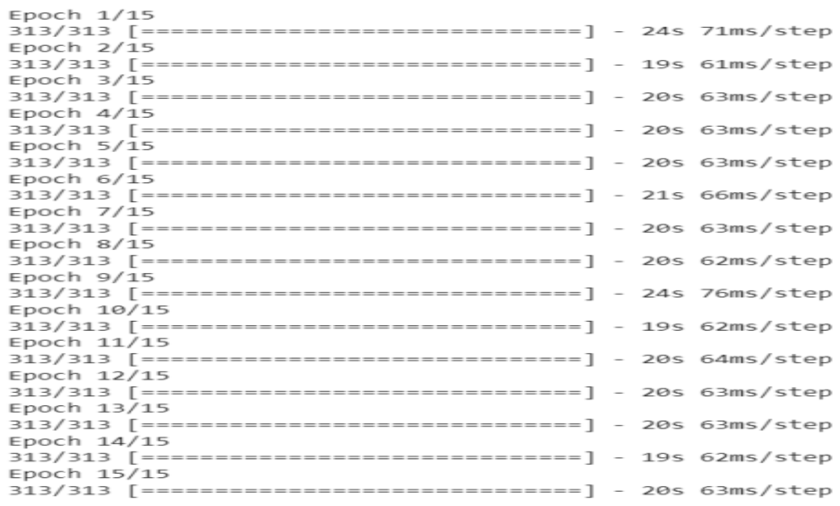
- Epoch = 15
- Batch Size = 32
- Learning rate = 0.01

Testing:

1. Final Accuracy: 8.5%
   For train:
   - Epoch 1: accuracy: 0.1000
   - Epoch 2: accuracy: 0.1001
   - Epoch 3: accuracy: 0.1001
   - Epoch 4: accuracy: 0.1001
   - Epoch 5: accuracy: 0.1001

   For test:
   - Accuracy: 0.0850

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 24s 71ms/step
Epoch 2/15
313/313 [==============================] - 19s 61ms/step
Epoch 3/15
313/313 [==============================] - 20s 63ms/step
Epoch 4/15
313/313 [==============================] - 20s 63ms/step
Epoch 5/15
313/313 [==============================] - 20s 63ms/step
Epoch 6/15
313/313 [==============================] - 21s 66ms/step
Epoch 7/15
313/313 [==============================] - 20s 63ms/step
Epoch 8/15
313/313 [==============================] - 20s 62ms/step
Epoch 9/15
313/313 [==============================] - 24s 76ms/step
Epoch 10/15
313/313 [==============================] - 19s 62ms/step
Epoch 11/15
313/313 [==============================] - 20s 64ms/step
Epoch 12/15
313/313 [==============================] - 20s 63ms/step
Epoch 13/15
313/313 [==============================] - 20s 63ms/step
Epoch 14/15
313/313 [==============================] - 19s 62ms/step
Epoch 15/15
313/313 [==============================] - 20s 63ms/step
```

4. Average time for Test = 1s 16ms/step

Model 5:

- Epoch = 15
- Batch Size = 32
- Learning rate = 0.0001

Testing:

1. Final Accuracy: 95.80%

   For train:

   Epoch 1: accuracy: 0.7959

   Epoch 2: accuracy: 0.9345

   Epoch 3: accuracy: 0.9571

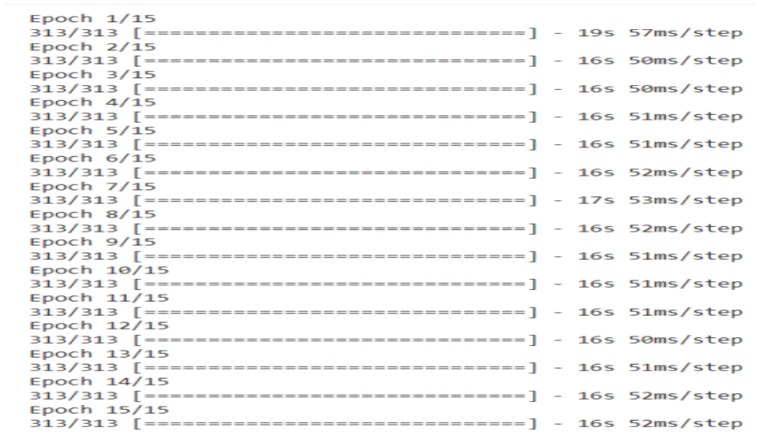
   Epoch 4: accuracy: 0.9696

   Epoch 5: accuracy: 0.9776

   For test:

   - Accuracy: 0.9580

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 19s 57ms/step
Epoch 2/15
313/313 [==============================] - 16s 50ms/step
Epoch 3/15
313/313 [==============================] - 16s 50ms/step
Epoch 4/15
313/313 [==============================] - 16s 51ms/step
Epoch 5/15
313/313 [==============================] - 16s 51ms/step
Epoch 6/15
313/313 [==============================] - 16s 52ms/step
Epoch 7/15
313/313 [==============================] - 17s 53ms/step
Epoch 8/15
313/313 [==============================] - 16s 52ms/step
Epoch 9/15
313/313 [==============================] - 16s 51ms/step
Epoch 10/15
313/313 [==============================] - 16s 51ms/step
Epoch 11/15
313/313 [==============================] - 16s 51ms/step
Epoch 12/15
313/313 [==============================] - 16s 50ms/step
Epoch 13/15
313/313 [==============================] - 16s 51ms/step
Epoch 14/15
313/313 [==============================] - 16s 52ms/step
Epoch 15/15
313/313 [==============================] - 16s 52ms/step
```

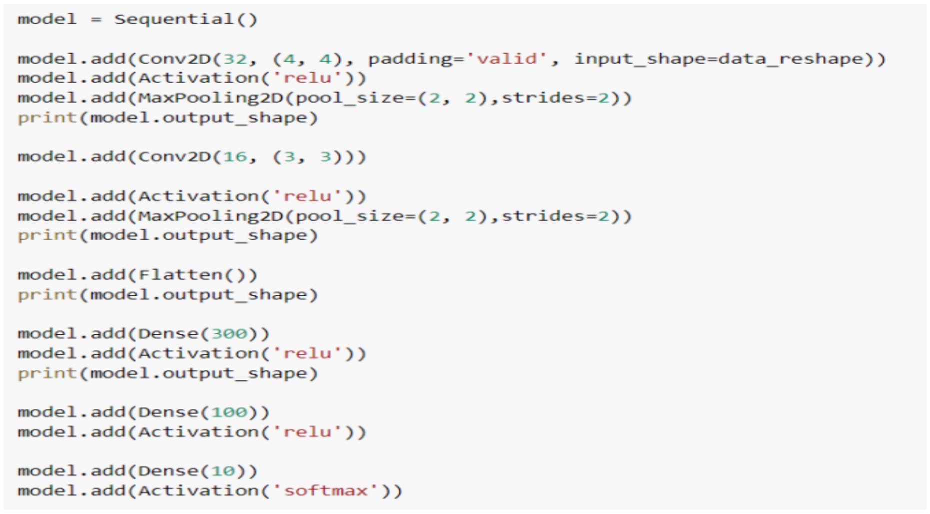
4. Average time for Test = 1s 12ms/step

Conclusion:
When using Learning Rate = 0.01 we got accuracy = 8.5% (model 4)
and when using Learning Rate = 0.0001 the accuracy was 95.80% (model 5)
and when using Learning Rate = 0.001 the accuracy was 98.5 % (model 3)
the higher the Learning Rate, the higher the accuracy
So, the best case was with Learning Rate = 0.001

The effect of changing the model Layers:

Model 6:

- Epoch = 15
- Batch Size = 32



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))

model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)

model.add(Dense(100))
model.add(Activation('relu'))

model.add(Dense(10))
model.add(Activation('softmax'))
```

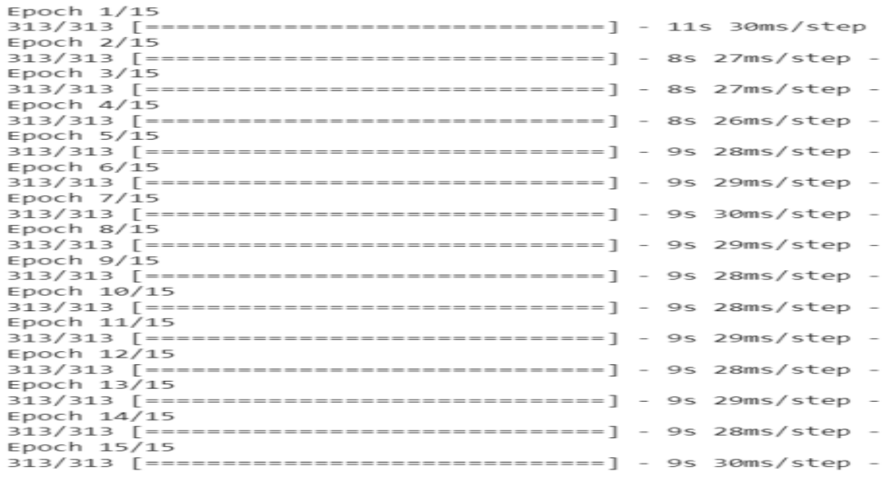
Testing:

1. Final Accuracy: 97.1%
   For train:
   - Epoch 1: accuracy: 0.7211
   - Epoch 2: accuracy: 0.9351
   - Epoch 3: accuracy: 0.9587
   - Epoch 4: accuracy: 0.9692
   - Epoch 5: accuracy: 0.9770

   For test:
   - Accuracy: 0.9710

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   FC_hidden1= $(5\ X\ 5\ X\ 16)\ X\ 300 + 300 = \ 120300$
   FC_hidden2 = $300\ X\ 100 + 100 = 30100$
   FC_output = $100\ X\ 10 + 10 = 1010$
   Number of Parameters = 156578 parameters

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 11s 30ms/step
Epoch 2/15
313/313 [==============================] - 8s 27ms/step -
Epoch 3/15
313/313 [==============================] - 8s 27ms/step -
Epoch 4/15
313/313 [==============================] - 8s 26ms/step -
Epoch 5/15
313/313 [==============================] - 9s 28ms/step -
Epoch 6/15
313/313 [==============================] - 9s 29ms/step -
Epoch 7/15
313/313 [==============================] - 9s 30ms/step -
Epoch 8/15
313/313 [==============================] - 9s 29ms/step -
Epoch 9/15
313/313 [==============================] - 9s 28ms/step -
Epoch 10/15
313/313 [==============================] - 9s 28ms/step -
Epoch 11/15
313/313 [==============================] - 9s 29ms/step -
Epoch 12/15
313/313 [==============================] - 9s 28ms/step -
Epoch 13/15
313/313 [==============================] - 9s 29ms/step -
Epoch 14/15
313/313 [==============================] - 9s 28ms/step -
Epoch 15/15
313/313 [==============================] - 9s 30ms/step -
```

4. Average time for Test = 1s 12ms/step

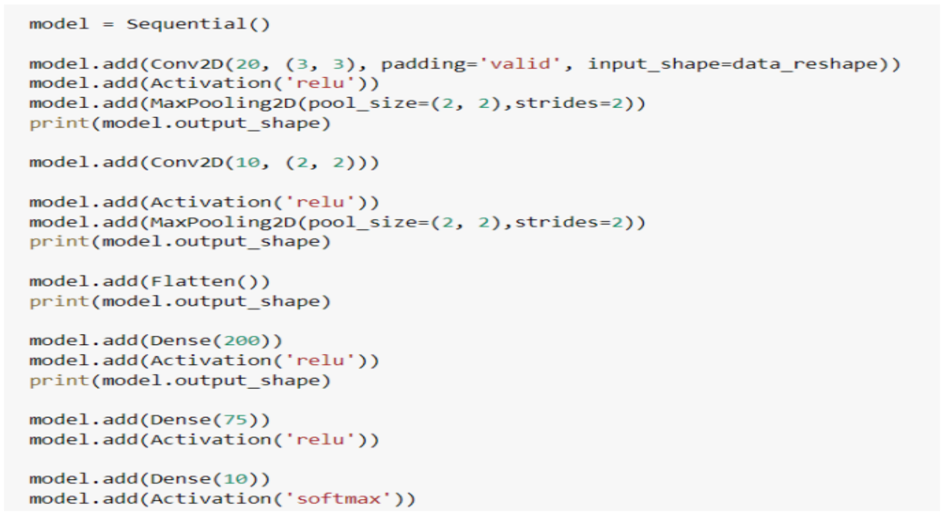
5. Conv1: Filters=32, kernel Size= 4 x 4 , activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_hidden2: Size=100, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

Model 7:

- Epoch = 15
- Batch Size = 32



```
model = Sequential()

model.add(Conv2D(20, (3, 3), padding='valid', input_shape=data_reshape))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(10, (2, 2)))

model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(200))
model.add(Activation('relu'))
print(model.output_shape)

model.add(Dense(75))
model.add(Activation('relu'))

model.add(Dense(10))
model.add(Activation('softmax'))
```

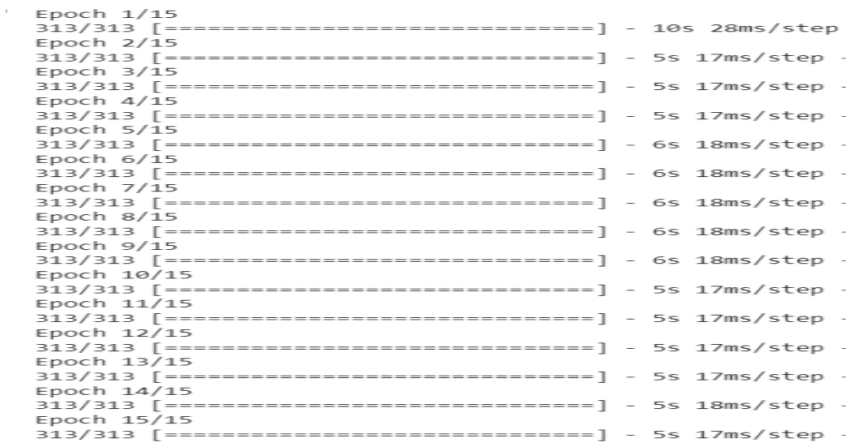
Testing:

1. Final Accuracy: 93.9%
   For train:
   - Epoch 1: accuracy: 0.4444
   - Epoch 2: accuracy: 0.8749
   - Epoch 3: accuracy: 0.9142
   - Epoch 4: accuracy: 0.9367
   - Epoch 5: accuracy: 0.9498

   For test:
   - Accuracy: 0.9390

2. Conv1 = $(3\ X\ 3\ X\ 1)20 + 20 = 200$
   Conv2 = $(2\ X\ 2\ X\ 20)\ 10 + 10 = 810$
   FC_hidden1= $(6\ X\ 6\ X\ 10)\ X\ 200 + 200 =\ 72200$
   FC_hidden2 = $200\ X\ 75 + 75 = 15075$
   FC_output = $75\ X\ 10 + 10 = 760$
   Number of Parameters = 89045 parameters

3. Average time for Train:

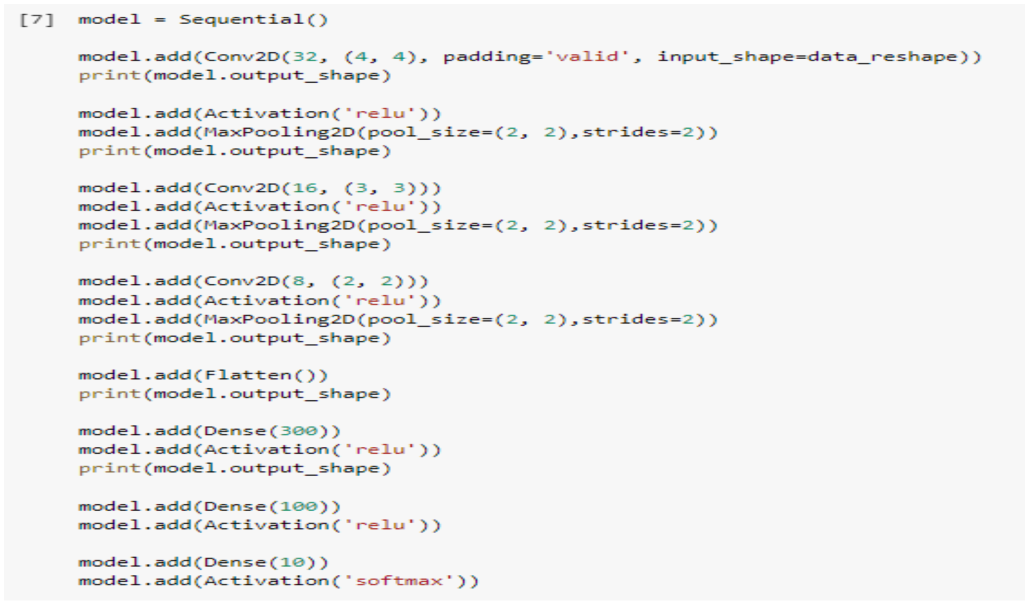


```
Epoch 1/15
313/313 [==============================] - 10s 28ms/step
Epoch 2/15
313/313 [==============================] - 5s 17ms/step -
Epoch 3/15
313/313 [==============================] - 5s 17ms/step -
Epoch 4/15
313/313 [==============================] - 5s 17ms/step -
Epoch 5/15
313/313 [==============================] - 6s 18ms/step -
Epoch 6/15
313/313 [==============================] - 6s 18ms/step -
Epoch 7/15
313/313 [==============================] - 6s 18ms/step -
Epoch 8/15
313/313 [==============================] - 6s 18ms/step -
Epoch 9/15
313/313 [==============================] - 6s 18ms/step -
Epoch 10/15
313/313 [==============================] - 5s 17ms/step -
Epoch 11/15
313/313 [==============================] - 5s 17ms/step -
Epoch 12/15
313/313 [==============================] - 5s 17ms/step -
Epoch 13/15
313/313 [==============================] - 5s 17ms/step -
Epoch 14/15
313/313 [==============================] - 5s 18ms/step -
Epoch 15/15
313/313 [==============================] - 5s 17ms/step -
```

4. Average time for Test = 0s 7ms/step
5. Conv1: Filters=20, kernel Size= 3 x 3, activation function= “Relu”
   Conv2: Filters=10, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=200, activation function= “Relu”
   FC_hidden2: Size=75, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”
6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Model 8:

- Epoch = 15
- Batch Size = 32



```
[7]  model = Sequential()

     model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
     print(model.output_shape)

     model.add(Activation('relu'))
     model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
     print(model.output_shape)

     model.add(Conv2D(16, (3, 3)))
     model.add(Activation('relu'))
     model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
     print(model.output_shape)

     model.add(Conv2D(8, (2, 2)))
     model.add(Activation('relu'))
     model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
     print(model.output_shape)

     model.add(Flatten())
     print(model.output_shape)

     model.add(Dense(300))
     model.add(Activation('relu'))
     print(model.output_shape)

     model.add(Dense(100))
     model.add(Activation('relu'))

     model.add(Dense(10))
     model.add(Activation('softmax'))
```

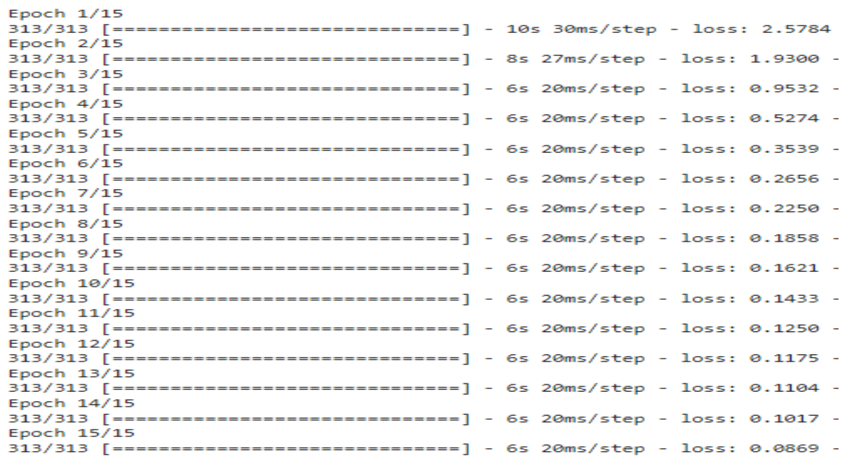
Testing:

1. Final Accuracy: 94.59999799728394%
   For train:
   - Epoch 1: accuracy: 0.1110
   - Epoch 2: accuracy: 0.3229
   - Epoch 3: accuracy: 0.7044
   - Epoch 4: accuracy: 0.8337
   - Epoch 5: accuracy: 0.8916

   For test:
   - accuracy: 0.9460

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 =\ 9900$
   FC_hidden2 = $300\ X\ 100 + 100 = 30010$
   FC_output = $100\ X\ 10 + 10 = 1010$
   Number of Parameters = 46608 parameters

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 10s 30ms/step - loss: 2.5784
Epoch 2/15
313/313 [==============================] - 8s 27ms/step - loss: 1.9300 -
Epoch 3/15
313/313 [==============================] - 6s 20ms/step - loss: 0.9532 -
Epoch 4/15
313/313 [==============================] - 6s 20ms/step - loss: 0.5274 -
Epoch 5/15
313/313 [==============================] - 6s 20ms/step - loss: 0.3539 -
Epoch 6/15
313/313 [==============================] - 6s 20ms/step - loss: 0.2656 -
Epoch 7/15
313/313 [==============================] - 6s 20ms/step - loss: 0.2250 -
Epoch 8/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1858 -
Epoch 9/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1621 -
Epoch 10/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1433 -
Epoch 11/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1250 -
Epoch 12/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1175 -
Epoch 13/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1104 -
Epoch 14/15
313/313 [==============================] - 6s 20ms/step - loss: 0.1017 -
Epoch 15/15
313/313 [==============================] - 6s 20ms/step - loss: 0.0869 -
```

4. Average time for Test = 0s 7ms/step

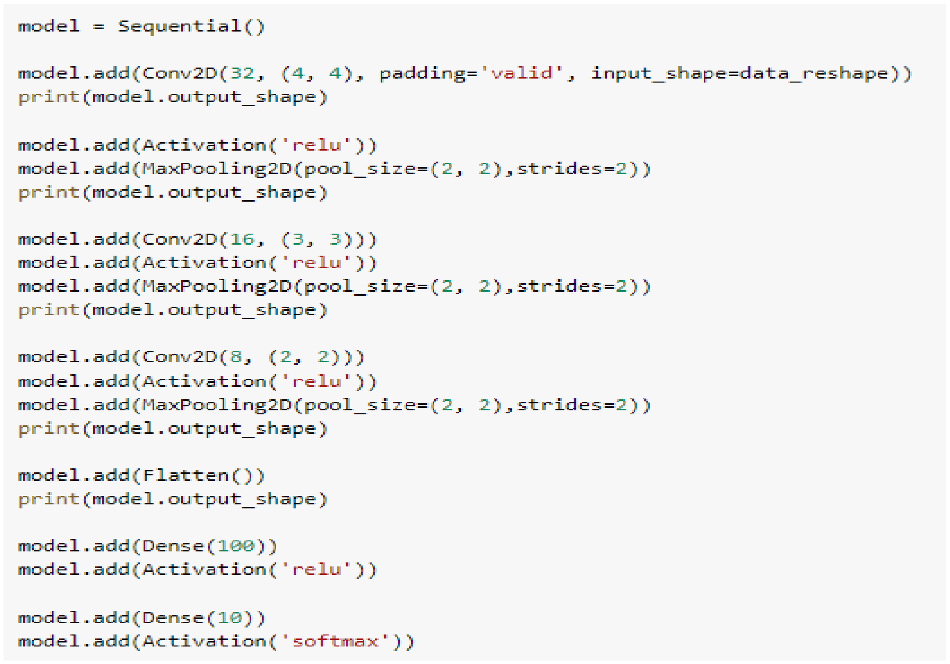
5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_hidden2: Size=100, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

Model 9:

- Epoch = 15
- Batch Size = 32



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(100))
model.add(Activation('relu'))

model.add(Dense(10))
model.add(Activation('softmax'))
```

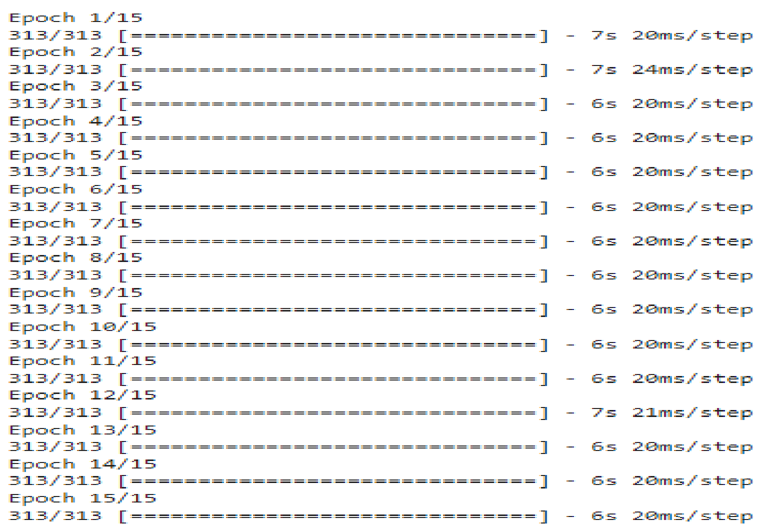
Testing:

1. Final Accuracy: 12.600%
   For train:
   - Epoch 1: accuracy: 0.1110
   - Epoch 2: accuracy: 0.1127
   - Epoch 3: accuracy: 0.1127
   - Epoch 4: accuracy: 0.1127
   - Epoch 5: accuracy: 0.1127

   For test:
   - accuracy: 0.1260

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 100 + 100 =\ 3300$
   FC_output = $100\ X\ 10 + 10 = 1010$
   Number of Parameters = 9998 parameters

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 7s 20ms/step
Epoch 2/15
313/313 [==============================] - 7s 24ms/step
Epoch 3/15
313/313 [==============================] - 6s 20ms/step
Epoch 4/15
313/313 [==============================] - 6s 20ms/step
Epoch 5/15
313/313 [==============================] - 6s 20ms/step
Epoch 6/15
313/313 [==============================] - 6s 20ms/step
Epoch 7/15
313/313 [==============================] - 6s 20ms/step
Epoch 8/15
313/313 [==============================] - 6s 20ms/step
Epoch 9/15
313/313 [==============================] - 6s 20ms/step
Epoch 10/15
313/313 [==============================] - 6s 20ms/step
Epoch 11/15
313/313 [==============================] - 6s 20ms/step
Epoch 12/15
313/313 [==============================] - 7s 21ms/step
Epoch 13/15
313/313 [==============================] - 6s 20ms/step
Epoch 14/15
313/313 [==============================] - 6s 20ms/step
Epoch 15/15
313/313 [==============================] - 6s 20ms/step
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=100, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

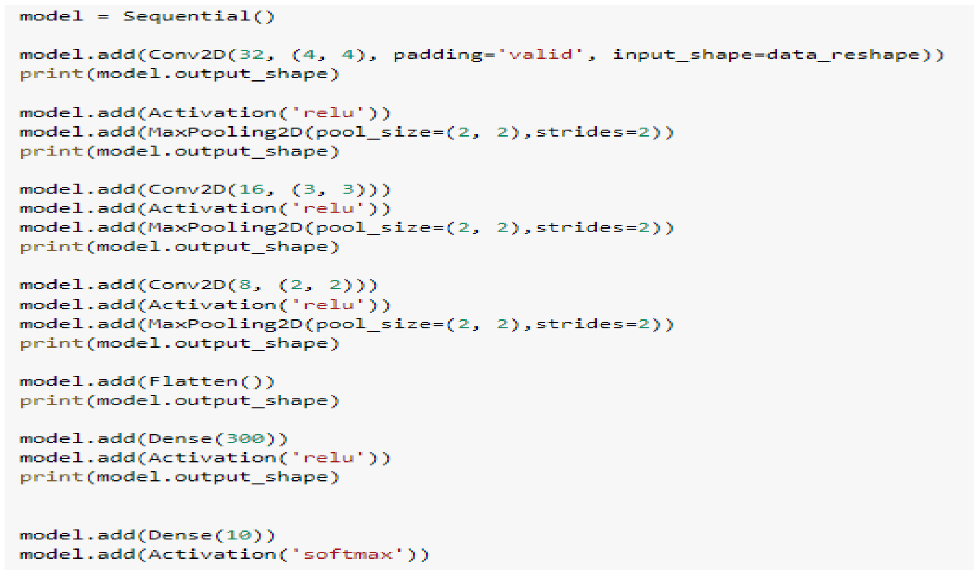
6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

____

Model 11:

- Epoch = 15
- Batch Size = 32



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

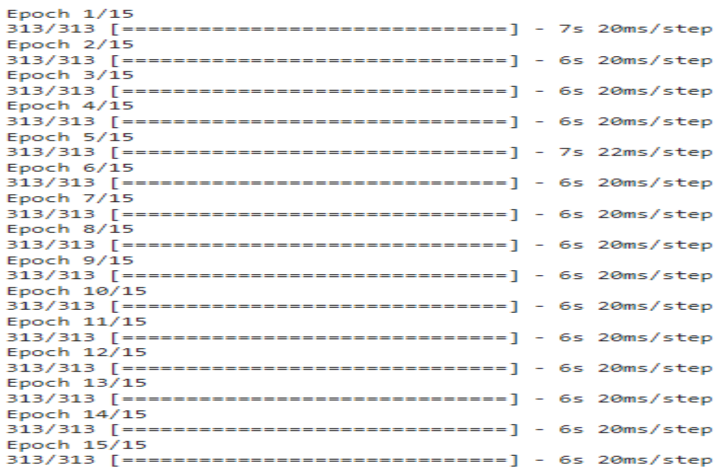
Testing:

1. Final Accuracy: 94.19999%
   For train:
   - Epoch 1: accuracy: 0.5675
   - Epoch 2: accuracy: 0.8474
   - Epoch 3: accuracy: 0.9116
   - Epoch 4: accuracy: 0.9325
   - Epoch 5: accuracy: 0.9433

   For test:
   - accuracy: 0.9420

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
313/313 [==============================] - 7s 20ms/step
Epoch 2/15
313/313 [==============================] - 6s 20ms/step
Epoch 3/15
313/313 [==============================] - 6s 20ms/step
Epoch 4/15
313/313 [==============================] - 6s 20ms/step
Epoch 5/15
313/313 [==============================] - 7s 22ms/step
Epoch 6/15
313/313 [==============================] - 6s 20ms/step
Epoch 7/15
313/313 [==============================] - 6s 20ms/step
Epoch 8/15
313/313 [==============================] - 6s 20ms/step
Epoch 9/15
313/313 [==============================] - 6s 20ms/step
Epoch 10/15
313/313 [==============================] - 6s 20ms/step
Epoch 11/15
313/313 [==============================] - 6s 20ms/step
Epoch 12/15
313/313 [==============================] - 6s 20ms/step
Epoch 13/15
313/313 [==============================] - 6s 20ms/step
Epoch 14/15
313/313 [==============================] - 6s 20ms/step
Epoch 15/15
313/313 [==============================] - 6s 20ms/step
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Conclusion:

In Model 3 we got Accuracy =98.5%with parameters =404868(higher accuracy but with higher parameters)
but in Model 11 the Accuracy was 94.199% with parameters = 18598(less parameters)

Best case is model 11: Accuracy =94.199% with parameters = 18598

Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
FC_hidden1: Size=300, activation function= “Relu”
FC_output: Size=10, activation function= “SoftMax”

The effect of changing the Batch size:

Model 12:

- Epoch = 15
- Batch Size = 64

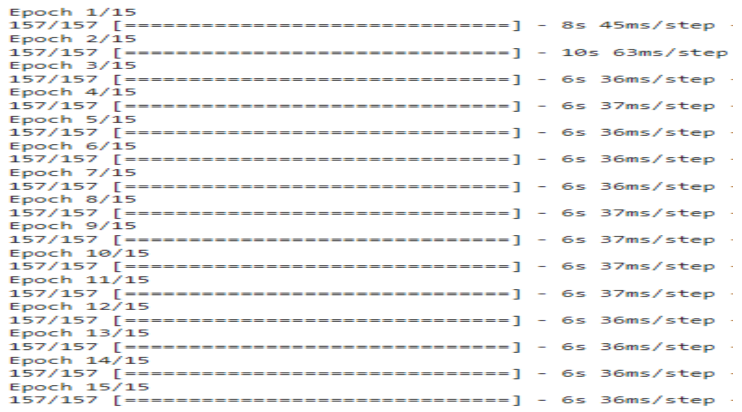
Testing:

1. Final Accuracy: 91.100%
   For train:
   - Epoch 1: accuracy: 0.1654
   - Epoch 2: accuracy: 0.2492
   - Epoch 3: accuracy: 0.3563
   - Epoch 4: accuracy: 0.5308
   - Epoch 5: accuracy: 0.7092

   For test:
   - accuracy: 0.9110

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
157/157 [==============================] - 8s 45ms/step
Epoch 2/15
157/157 [==============================] - 10s 63ms/step
Epoch 3/15
157/157 [==============================] - 6s 36ms/step
Epoch 4/15
157/157 [==============================] - 6s 37ms/step
Epoch 5/15
157/157 [==============================] - 6s 36ms/step
Epoch 6/15
157/157 [==============================] - 6s 36ms/step
Epoch 7/15
157/157 [==============================] - 6s 36ms/step
Epoch 8/15
157/157 [==============================] - 6s 37ms/step
Epoch 9/15
157/157 [==============================] - 6s 37ms/step
Epoch 10/15
157/157 [==============================] - 6s 37ms/step
Epoch 11/15
157/157 [==============================] - 6s 37ms/step
Epoch 12/15
157/157 [==============================] - 6s 36ms/step
Epoch 13/15
157/157 [==============================] - 6s 36ms/step
Epoch 14/15
157/157 [==============================] - 6s 36ms/step
Epoch 15/15
157/157 [==============================] - 6s 36ms/step
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

Model 13:

- Epoch = 15
- Batch Size = 96

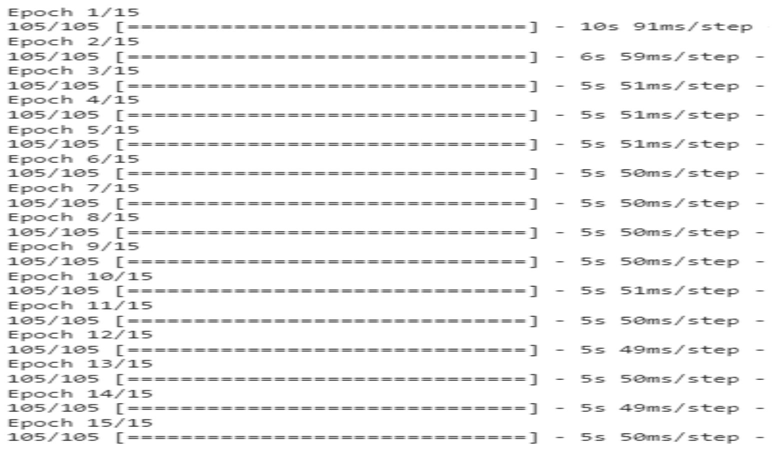
Testing:

1. Final Accuracy: 93.00%
   For train:
   - Epoch 1: accuracy: 0.2288
   - Epoch 2: accuracy: 0.3789
   - Epoch 3: accuracy: 0.4797
   - Epoch 4: accuracy: 0.6237
   - Epoch 5: accuracy: 0.8348

   For test:
   - accuracy: 0.9300

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 =\ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
105/105 [==============================] - 10s 91ms/step
Epoch 2/15
105/105 [==============================] - 6s 59ms/step -
Epoch 3/15
105/105 [==============================] - 5s 51ms/step -
Epoch 4/15
105/105 [==============================] - 5s 51ms/step -
Epoch 5/15
105/105 [==============================] - 5s 51ms/step -
Epoch 6/15
105/105 [==============================] - 5s 50ms/step -
Epoch 7/15
105/105 [==============================] - 5s 50ms/step -
Epoch 8/15
105/105 [==============================] - 5s 50ms/step -
Epoch 9/15
105/105 [==============================] - 5s 50ms/step -
Epoch 10/15
105/105 [==============================] - 5s 51ms/step -
Epoch 11/15
105/105 [==============================] - 5s 50ms/step -
Epoch 12/15
105/105 [==============================] - 5s 49ms/step -
Epoch 13/15
105/105 [==============================] - 5s 50ms/step -
Epoch 14/15
105/105 [==============================] - 5s 49ms/step -
Epoch 15/15
105/105 [==============================] - 5s 50ms/step -
```

4. Average time for Test =
   0s 8ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Model 14:

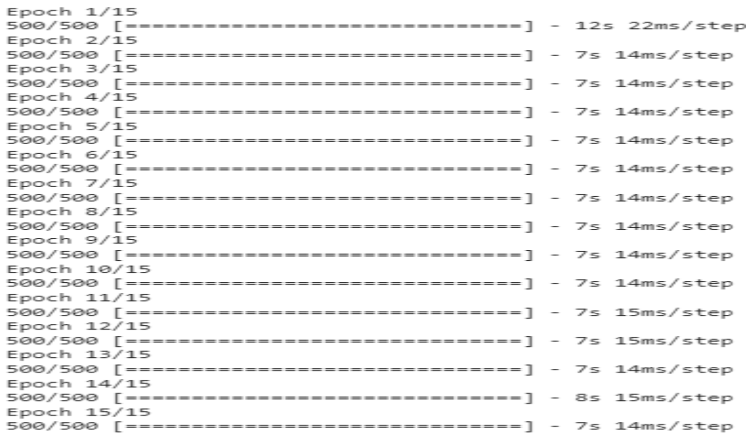
- Epoch = 15
- Batch Size = 20

Testing:

1. Final Accuracy: 94.30%
   For train:
   - Epoch 1: accuracy: 0.4604
   - Epoch 2: accuracy: 0.8454
   - Epoch 3: accuracy: 0.9004
   - Epoch 4: accuracy: 0.9221
   - Epoch 5: accuracy: 0.9345

   For test:
   - accuracy: 0.9430

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters
3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 12s 22ms/step
Epoch 2/15
500/500 [==============================] - 7s 14ms/step
Epoch 3/15
500/500 [==============================] - 7s 14ms/step
Epoch 4/15
500/500 [==============================] - 7s 14ms/step
Epoch 5/15
500/500 [==============================] - 7s 14ms/step
Epoch 6/15
500/500 [==============================] - 7s 14ms/step
Epoch 7/15
500/500 [==============================] - 7s 14ms/step
Epoch 8/15
500/500 [==============================] - 7s 14ms/step
Epoch 9/15
500/500 [==============================] - 7s 14ms/step
Epoch 10/15
500/500 [==============================] - 7s 14ms/step
Epoch 11/15
500/500 [==============================] - 7s 15ms/step
Epoch 12/15
500/500 [==============================] - 7s 15ms/step
Epoch 13/15
500/500 [==============================] - 7s 14ms/step
Epoch 14/15
500/500 [==============================] - 8s 15ms/step
Epoch 15/15
500/500 [==============================] - 7s 14ms/step
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

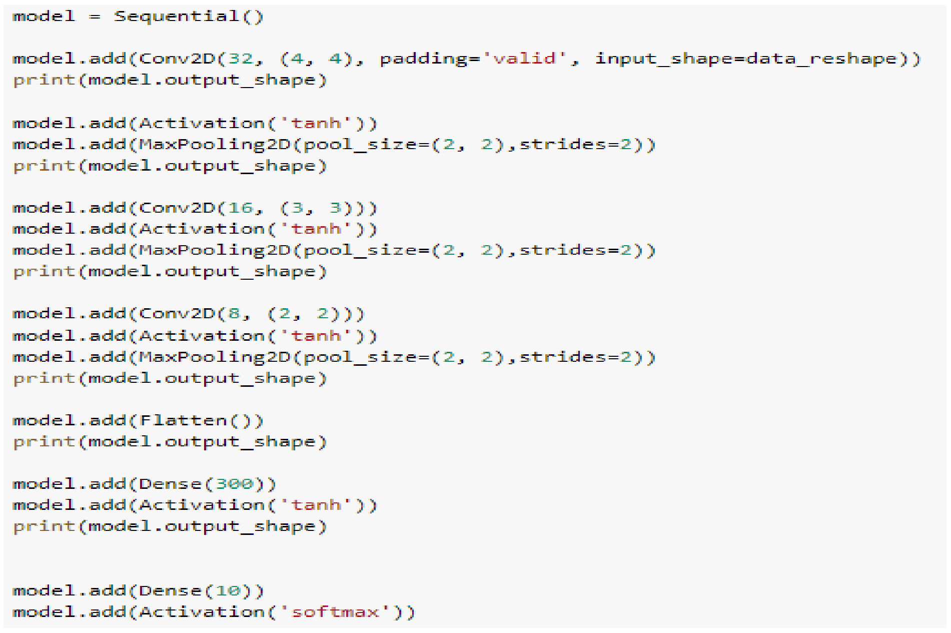
7. Optimizer: SGD with learning rate and momentum.

Conclusion:
When using Batch Size = 64 we got accuracy = 91.1% (model 12)
and when using Batch Size = 96 the accuracy was 93% (model 13)
and when using Learning Rate = 20 the accuracy was 94.3 % (model 14)
the lower the Batch Size, the higher the accuracy
So, the best case was in model 14 with Batch Size =20 and gives Accuracy =94.3%

The effect of changing the Activation Functions:

Model 15:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('tanh'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('tanh'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('tanh'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('tanh'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 94.1%

   For train:
   - Epoch 1: accuracy: 0.5373
   - Epoch 2: accuracy: 0.8590
   - Epoch 3: accuracy: 0.9082
   - Epoch 4: accuracy: 0.9241
   - Epoch 5: accuracy: 0.9346

   For test:
   - accuracy: 0.9400

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$

   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$

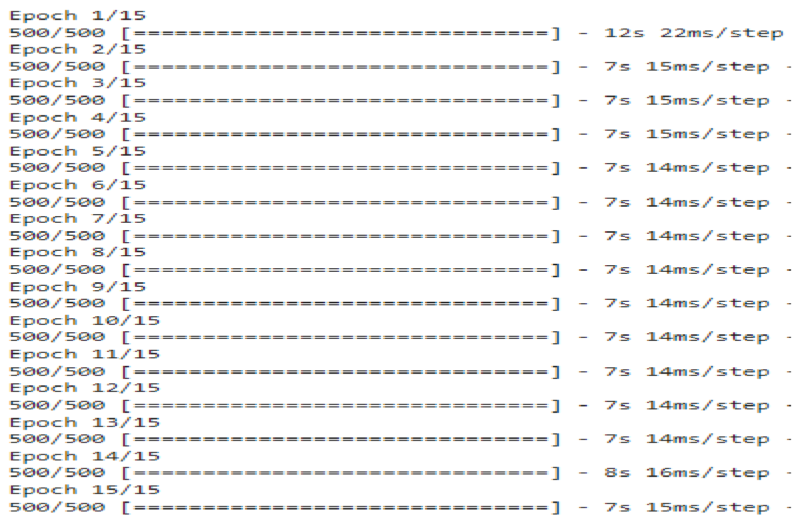
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$

   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = 9900$

   FC_output = $300X\ 10 + 10 = 3010$

   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 12s 22ms/step
Epoch 2/15
500/500 [==============================] - 7s 15ms/step -
Epoch 3/15
500/500 [==============================] - 7s 15ms/step -
Epoch 4/15
500/500 [==============================] - 7s 15ms/step -
Epoch 5/15
500/500 [==============================] - 7s 14ms/step -
Epoch 6/15
500/500 [==============================] - 7s 14ms/step -
Epoch 7/15
500/500 [==============================] - 7s 14ms/step -
Epoch 8/15
500/500 [==============================] - 7s 14ms/step -
Epoch 9/15
500/500 [==============================] - 7s 14ms/step -
Epoch 10/15
500/500 [==============================] - 7s 14ms/step -
Epoch 11/15
500/500 [==============================] - 7s 14ms/step -
Epoch 12/15
500/500 [==============================] - 7s 14ms/step -
Epoch 13/15
500/500 [==============================] - 7s 14ms/step -
Epoch 14/15
500/500 [==============================] - 8s 16ms/step -
Epoch 15/15
500/500 [==============================] - 7s 15ms/step -
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Tanh”

   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Tanh”

   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Tanh”

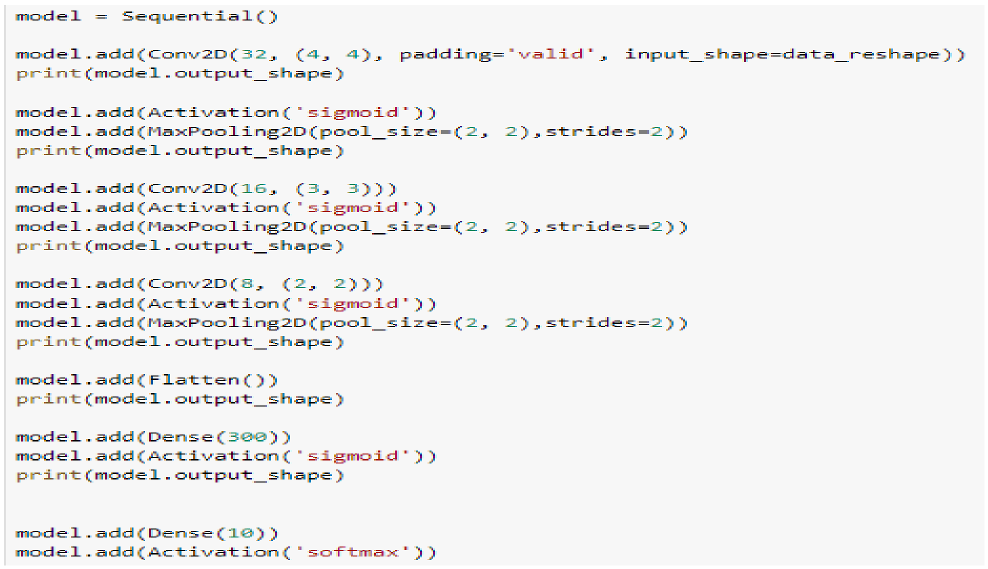
   FC_hidden1: Size=300, activation function= “Tanh”

   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Model 16:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('sigmoid'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('sigmoid'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('sigmoid'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('sigmoid'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

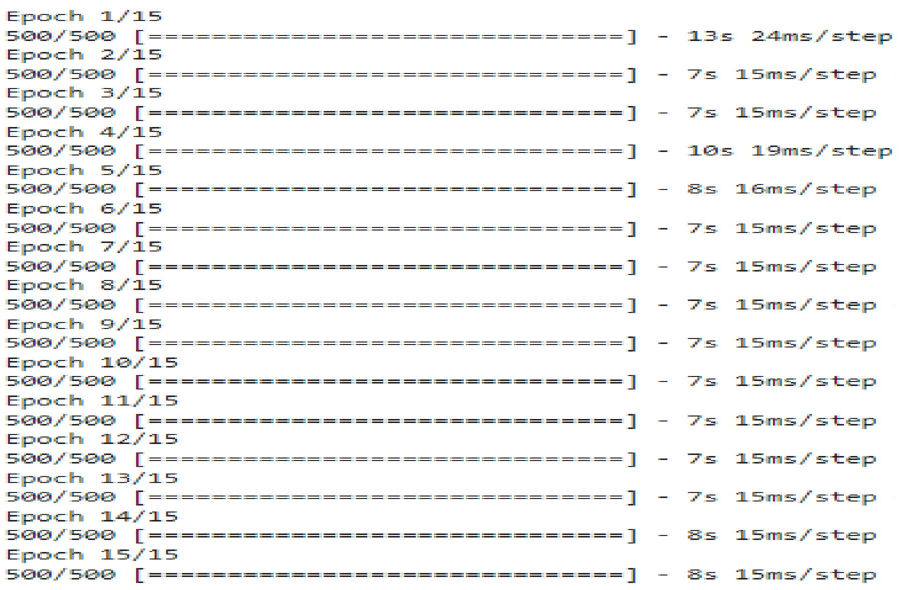
Testing:

1. Final Accuracy: 10.700%
   For train:
   - Epoch 1: accuracy: 0.1092
   - Epoch 2: accuracy: 0.0974
   - Epoch 3: accuracy: 0.1039
   - Epoch 4: accuracy: 0.1103
   - Epoch 5: accuracy: 0.1045

   For test:
   - accuracy 0.1070

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 13s 24ms/step
Epoch 2/15
500/500 [==============================] - 7s 15ms/step
Epoch 3/15
500/500 [==============================] - 7s 15ms/step
Epoch 4/15
500/500 [==============================] - 10s 19ms/step
Epoch 5/15
500/500 [==============================] - 8s 16ms/step
Epoch 6/15
500/500 [==============================] - 7s 15ms/step
Epoch 7/15
500/500 [==============================] - 7s 15ms/step
Epoch 8/15
500/500 [==============================] - 7s 15ms/step
Epoch 9/15
500/500 [==============================] - 7s 15ms/step
Epoch 10/15
500/500 [==============================] - 7s 15ms/step
Epoch 11/15
500/500 [==============================] - 7s 15ms/step
Epoch 12/15
500/500 [==============================] - 7s 15ms/step
Epoch 13/15
500/500 [==============================] - 7s 15ms/step
Epoch 14/15
500/500 [==============================] - 8s 15ms/step
Epoch 15/15
500/500 [==============================] - 8s 15ms/step
```

4. Average time for Test = 0s 7ms/step

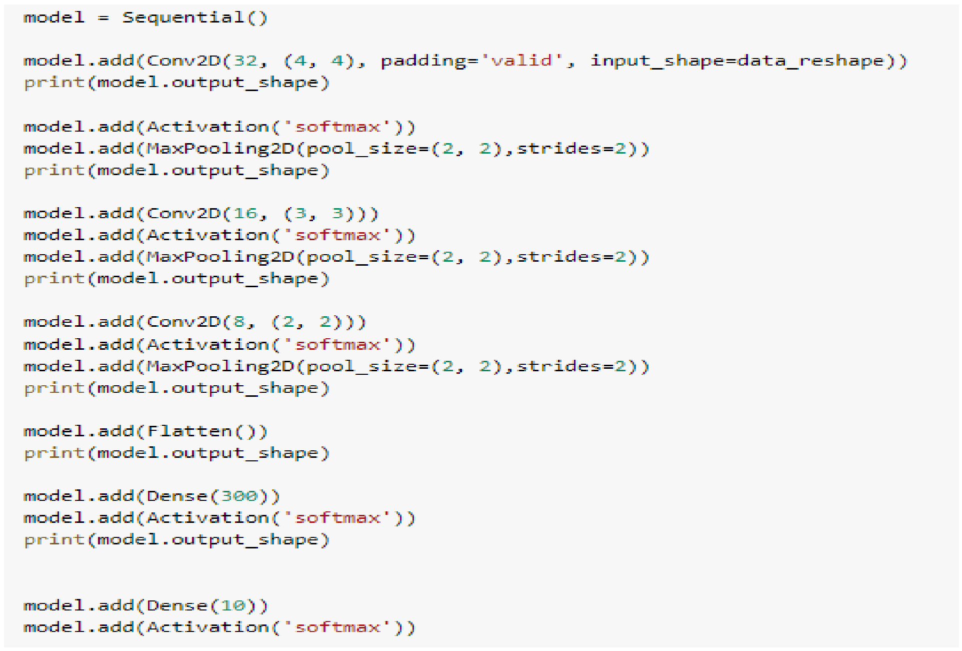
5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Sigmoid”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Sigmoid”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Sigmoid”
   FC_hidden1: Size=300, activation function= “Sigmoid”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9

7. Optimizer: SGD with learning rate and momentum.

Model 17:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('softmax'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('softmax'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('softmax'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('softmax'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 12.60%

   For train:
   - Epoch 1: accuracy: 0.1105
   - Epoch 2: accuracy: 0.1127
   - Epoch 3: accuracy: 0.1127
   - Epoch 4: accuracy: 0.1127
   - Epoch 5: accuracy: 0.1127

   For test:
   - accuracy: 0.1260

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$

   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$

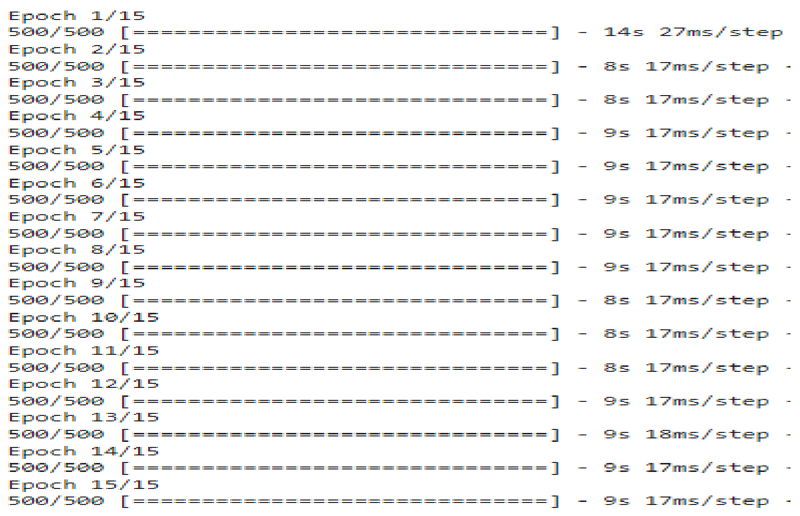
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$

   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$

   FC_output = $300X\ 10 + 10 = 3010$

   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 14s 27ms/step
Epoch 2/15
500/500 [==============================] - 8s 17ms/step -
Epoch 3/15
500/500 [==============================] - 8s 17ms/step -
Epoch 4/15
500/500 [==============================] - 9s 17ms/step -
Epoch 5/15
500/500 [==============================] - 9s 17ms/step -
Epoch 6/15
500/500 [==============================] - 9s 17ms/step -
Epoch 7/15
500/500 [==============================] - 9s 17ms/step -
Epoch 8/15
500/500 [==============================] - 9s 17ms/step -
Epoch 9/15
500/500 [==============================] - 8s 17ms/step -
Epoch 10/15
500/500 [==============================] - 8s 17ms/step -
Epoch 11/15
500/500 [==============================] - 8s 17ms/step -
Epoch 12/15
500/500 [==============================] - 9s 17ms/step -
Epoch 13/15
500/500 [==============================] - 9s 18ms/step -
Epoch 14/15
500/500 [==============================] - 9s 17ms/step -
Epoch 15/15
500/500 [==============================] - 9s 17ms/step -
```

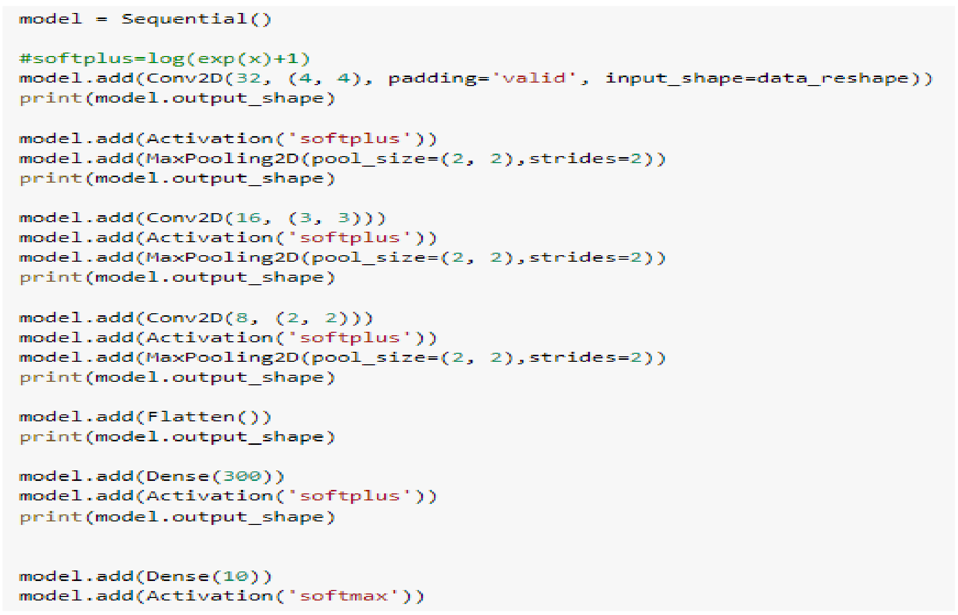
4. Average time for Test = 0s 9ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “SoftMax”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “SoftMax”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “SoftMax”
   FC_hidden1: Size=300, activation function= “SoftMax”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Model 18:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

#softplus=log(exp(x)+1)
model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)

model.add(Activation('softplus'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('softplus'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(8, (2, 2)))
model.add(Activation('softplus'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('softplus'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 11.59 %

   For train:
   - Epoch 1: accuracy: 0.1021
   - Epoch 2: accuracy: 0.1042
   - Epoch 3: accuracy: 0.1036
   - Epoch 4: accuracy: 0.1006
   - Epoch 5: accuracy: 0.1014

   For test:
   - accuracy: 0.1160

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$

   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$

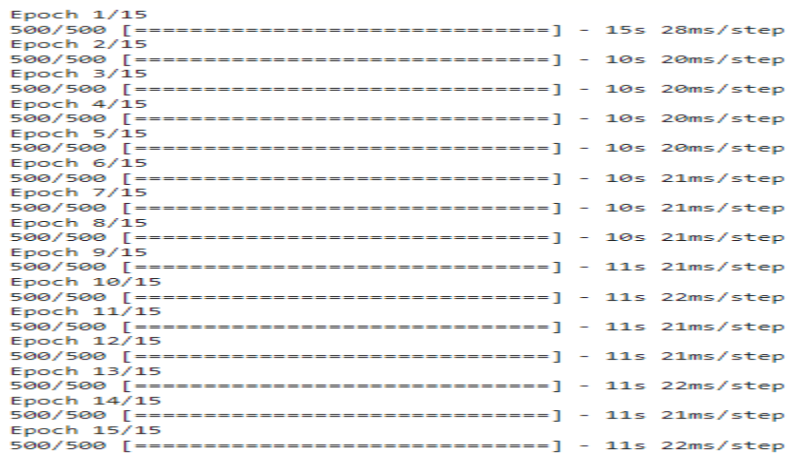
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$

   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 =\ 9900$

   FC_output = $300X\ 10 + 10 = 3010$

   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 15s 28ms/step
Epoch 2/15
500/500 [==============================] - 10s 20ms/step
Epoch 3/15
500/500 [==============================] - 10s 20ms/step
Epoch 4/15
500/500 [==============================] - 10s 20ms/step
Epoch 5/15
500/500 [==============================] - 10s 20ms/step
Epoch 6/15
500/500 [==============================] - 10s 21ms/step
Epoch 7/15
500/500 [==============================] - 10s 21ms/step
Epoch 8/15
500/500 [==============================] - 10s 21ms/step
Epoch 9/15
500/500 [==============================] - 11s 21ms/step
Epoch 10/15
500/500 [==============================] - 11s 22ms/step
Epoch 11/15
500/500 [==============================] - 11s 21ms/step
Epoch 12/15
500/500 [==============================] - 11s 21ms/step
Epoch 13/15
500/500 [==============================] - 11s 22ms/step
Epoch 14/15
500/500 [==============================] - 11s 21ms/step
Epoch 15/15
500/500 [==============================] - 11s 22ms/step
```

4. Average time for Test = 0s 10ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Softplus”

   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Softplus”

   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Softplus”

   FC_hidden1: Size=300, activation function= “Softplus”

   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
   Optimizer: Momentum=0.9
7. Optimizer: SGD with learning rate and momentum.

Conclusion:
In model 15 we used activation function “Tanh” and got accuracy = 94.1%
and in model 16 we used “sigmoid” and got accuracy 10.7 %
and in model 17 we used “softmax” and got accuracy = 12.6 %
And in model 18 we used “softplus” and got accuracy = 11.59 %
Best Case: Using Activation Function “Relu” with Accuracy =94.30%

The effect of using different Optimizers:

Model 19:

- Epoch = 15
- Batch Size = 20

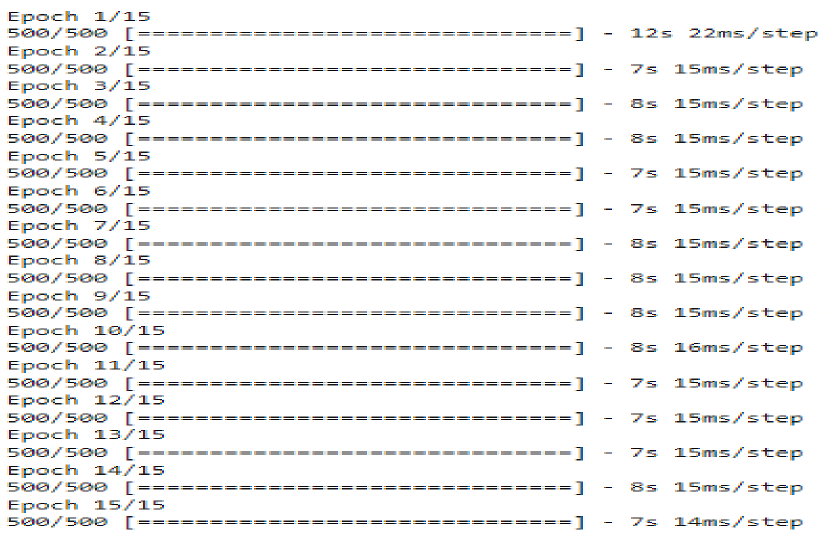
Testing:

1. Final Accuracy: 95.70%
   For train:
   - Epoch 1: accuracy: 0.7693
   - Epoch 2: accuracy: 0.9098
   - Epoch 3: accuracy: 0.9392
   - Epoch 4: accuracy: 0.9495
   - Epoch 5: accuracy: 0.9572

   For test:
   - accuracy: 0.9570

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 12s 22ms/step
Epoch 2/15
500/500 [==============================] - 7s 15ms/step
Epoch 3/15
500/500 [==============================] - 8s 15ms/step
Epoch 4/15
500/500 [==============================] - 8s 15ms/step
Epoch 5/15
500/500 [==============================] - 7s 15ms/step
Epoch 6/15
500/500 [==============================] - 7s 15ms/step
Epoch 7/15
500/500 [==============================] - 8s 15ms/step
Epoch 8/15
500/500 [==============================] - 8s 15ms/step
Epoch 9/15
500/500 [==============================] - 8s 15ms/step
Epoch 10/15
500/500 [==============================] - 8s 16ms/step
Epoch 11/15
500/500 [==============================] - 7s 15ms/step
Epoch 12/15
500/500 [==============================] - 7s 15ms/step
Epoch 13/15
500/500 [==============================] - 7s 15ms/step
Epoch 14/15
500/500 [==============================] - 8s 15ms/step
Epoch 15/15
500/500 [==============================] - 7s 14ms/step
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001

7. Optimizer: Adaptive Moment Estimation (Adam).

Model 20:

- Epoch = 15
- Batch Size = 20

Testing:

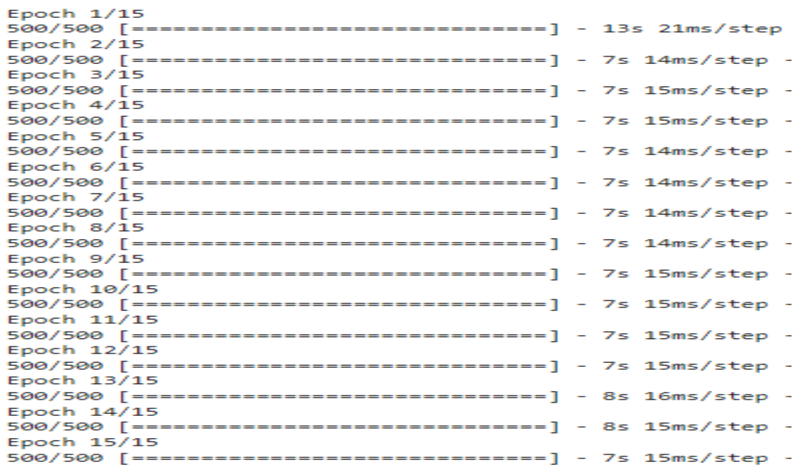
1. Final Accuracy: 94.900%
   For train:
   - Epoch 1: accuracy: 0.8192
   - Epoch 2: accuracy: 0.9279
   - Epoch 3: accuracy: 0.9475
   - Epoch 4: accuracy: 0.9572
   - Epoch 5: accuracy: 0.9660

For test:

- accuracy: 0.9490

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 =\ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 13s 21ms/step
Epoch 2/15
500/500 [==============================] - 7s 14ms/step -
Epoch 3/15
500/500 [==============================] - 7s 15ms/step -
Epoch 4/15
500/500 [==============================] - 7s 15ms/step -
Epoch 5/15
500/500 [==============================] - 7s 14ms/step -
Epoch 6/15
500/500 [==============================] - 7s 14ms/step -
Epoch 7/15
500/500 [==============================] - 7s 14ms/step -
Epoch 8/15
500/500 [==============================] - 7s 14ms/step -
Epoch 9/15
500/500 [==============================] - 7s 15ms/step -
Epoch 10/15
500/500 [==============================] - 7s 15ms/step -
Epoch 11/15
500/500 [==============================] - 7s 15ms/step -
Epoch 12/15
500/500 [==============================] - 7s 15ms/step -
Epoch 13/15
500/500 [==============================] - 8s 16ms/step -
Epoch 14/15
500/500 [==============================] - 8s 15ms/step -
Epoch 15/15
500/500 [==============================] - 7s 15ms/step -
```

4. Average time for Test = 0s 7ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001

7. Optimizer: Nadam (Nesterov-accelerated Adaptive Moment Estimation)

Conclusion:

In model 19 we used Adaptive Moment Estimation (Adam) and got accuracy = 95.7 %
and in model 20 we used Nadam (Nesterov-accelerated Adaptive Moment Estimation) and got accuracy = 94.9 %

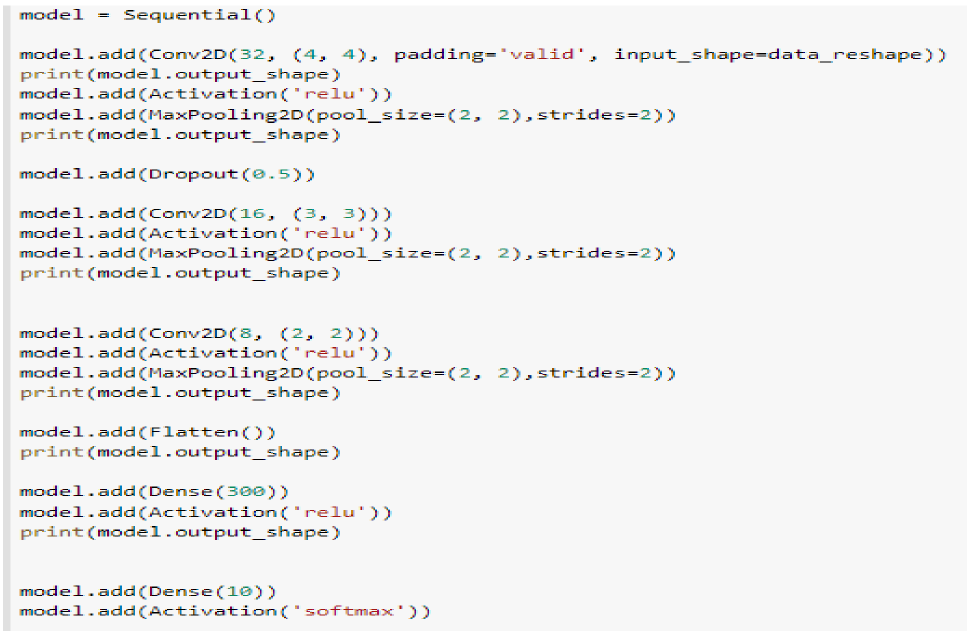
Best Case Using Optimizer: Adaptive Moment Estimation (Adam) With Accuracy = 95.70%

____

The effect of adding Drop out Layers:

Model 21:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Dropout(0.5))

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 96.89%

   For train:

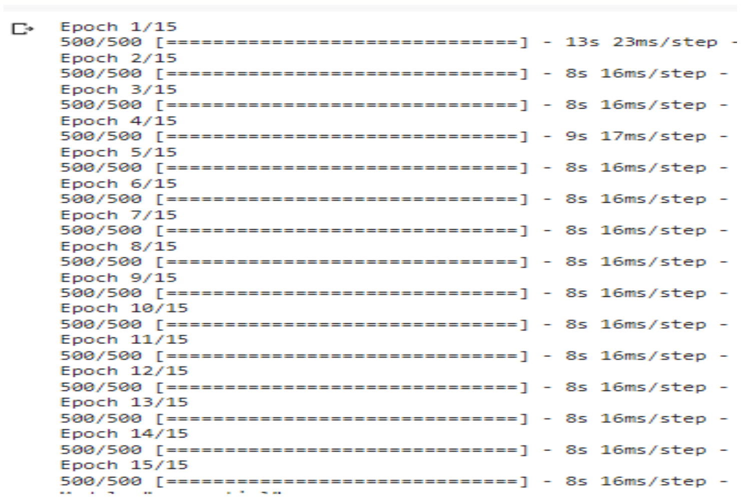
   - Epoch 1: accuracy: 0.4734
   - Epoch 2: accuracy: 0.8052
   - Epoch 3: accuracy: 0.8725
   - Epoch 4: accuracy: 0.8999
   - Epoch 5: accuracy: 0.9180

   For test:

   - Accuracy: 0.9690

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 13s 23ms/step -
Epoch 2/15
500/500 [==============================] - 8s 16ms/step -
Epoch 3/15
500/500 [==============================] - 8s 16ms/step -
Epoch 4/15
500/500 [==============================] - 9s 17ms/step -
Epoch 5/15
500/500 [==============================] - 8s 16ms/step -
Epoch 6/15
500/500 [==============================] - 8s 16ms/step -
Epoch 7/15
500/500 [==============================] - 8s 16ms/step -
Epoch 8/15
500/500 [==============================] - 8s 16ms/step -
Epoch 9/15
500/500 [==============================] - 8s 16ms/step -
Epoch 10/15
500/500 [==============================] - 8s 16ms/step -
Epoch 11/15
500/500 [==============================] - 8s 16ms/step -
Epoch 12/15
500/500 [==============================] - 8s 16ms/step -
Epoch 13/15
500/500 [==============================] - 8s 16ms/step -
Epoch 14/15
500/500 [==============================] - 8s 16ms/step -
Epoch 15/15
500/500 [==============================] - 8s 16ms/step -
```

4. Average time for Test = 0s 7ms/step

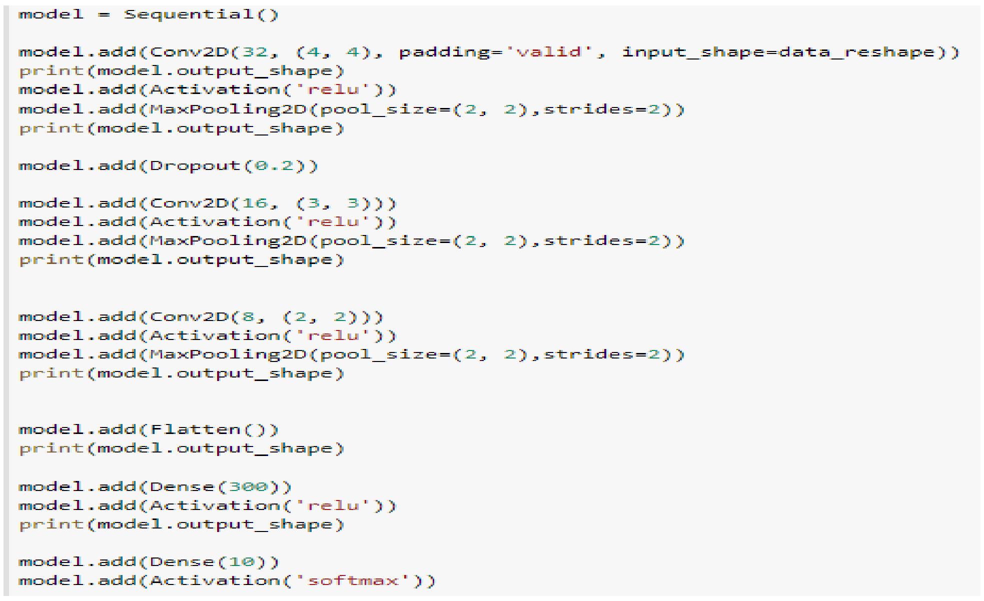
5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Drop out Layer with rate = 0.5
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001

7. Optimizer: Adaptive Moment Estimation (Adam).

Model 22:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Dropout(0.2))

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)

model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

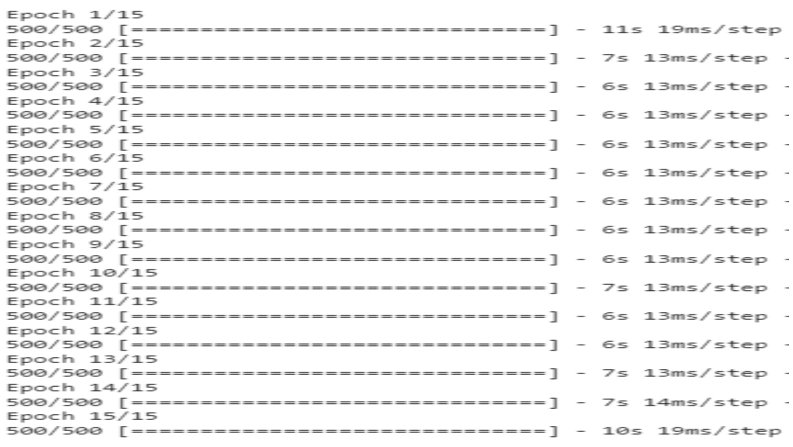
1. Final Accuracy: 96.79%

   For train:
   - Epoch 1: accuracy: 0.7055
   - Epoch 2: accuracy: 0.8947
   - Epoch 3: accuracy: 0.9275
   - Epoch 4: accuracy: 0.9444
   - Epoch 5: accuracy: 0.9542

   For test:
   - Accuracy: 0.9680

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 11s 19ms/step
Epoch 2/15
500/500 [==============================] - 7s 13ms/step -
Epoch 3/15
500/500 [==============================] - 6s 13ms/step -
Epoch 4/15
500/500 [==============================] - 6s 13ms/step -
Epoch 5/15
500/500 [==============================] - 6s 13ms/step -
Epoch 6/15
500/500 [==============================] - 6s 13ms/step -
Epoch 7/15
500/500 [==============================] - 6s 13ms/step -
Epoch 8/15
500/500 [==============================] - 6s 13ms/step -
Epoch 9/15
500/500 [==============================] - 6s 13ms/step -
Epoch 10/15
500/500 [==============================] - 7s 13ms/step -
Epoch 11/15
500/500 [==============================] - 6s 13ms/step -
Epoch 12/15
500/500 [==============================] - 6s 13ms/step -
Epoch 13/15
500/500 [==============================] - 7s 13ms/step -
Epoch 14/15
500/500 [==============================] - 7s 14ms/step -
Epoch 15/15
500/500 [==============================] - 10s 19ms/step
```

4. Average time for Test = 0s 5ms/step

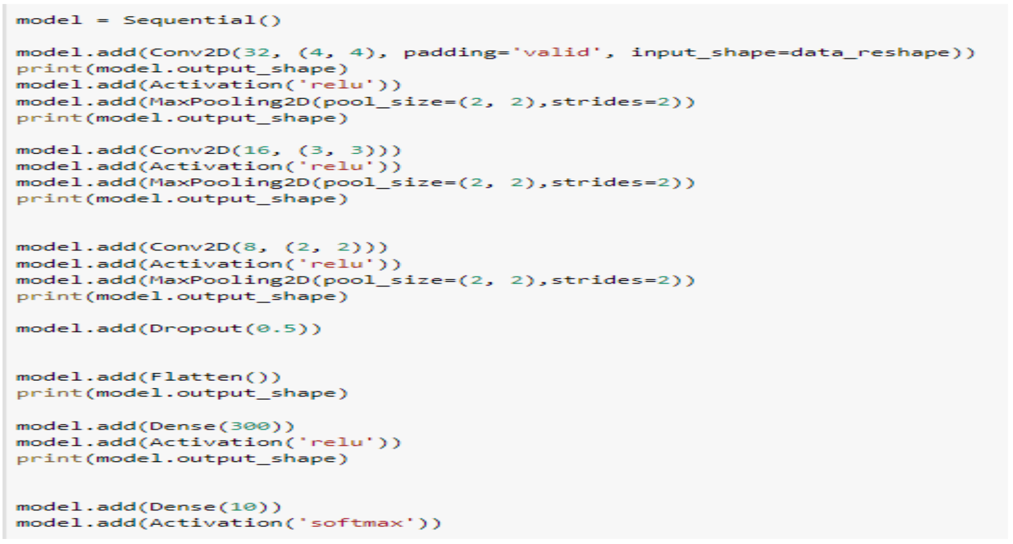
5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Drop out Layer with rate = 0.2
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
7. Optimizer: Adaptive Moment Estimation (Adam).

____

Model 23:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Dropout(0.5))


model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)


model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 92.599%

   For train:

   - Epoch 1: accuracy: 0.4217
   - Epoch 2: accuracy: 0.6671
   - Epoch 3: accuracy: 0.7390
   - Epoch 4: accuracy: 0.7611
   - Epoch 5: accuracy: 0.7897

   For test:

   - Accuracy: 0.92599

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$

   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$

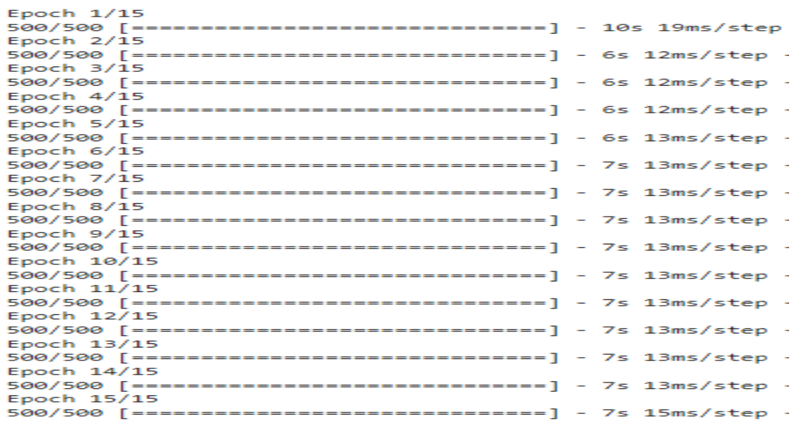
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$

   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 =\ 9900$

   FC_output = $300 X\ 10 + 10 = 3010$

   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 10s 19ms/step
Epoch 2/15
500/500 [==============================] - 6s 12ms/step -
Epoch 3/15
500/500 [==============================] - 6s 12ms/step -
Epoch 4/15
500/500 [==============================] - 6s 12ms/step -
Epoch 5/15
500/500 [==============================] - 6s 13ms/step -
Epoch 6/15
500/500 [==============================] - 7s 13ms/step -
Epoch 7/15
500/500 [==============================] - 7s 13ms/step -
Epoch 8/15
500/500 [==============================] - 7s 13ms/step -
Epoch 9/15
500/500 [==============================] - 7s 13ms/step -
Epoch 10/15
500/500 [==============================] - 7s 13ms/step -
Epoch 11/15
500/500 [==============================] - 7s 13ms/step -
Epoch 12/15
500/500 [==============================] - 7s 13ms/step -
Epoch 13/15
500/500 [==============================] - 7s 13ms/step -
Epoch 14/15
500/500 [==============================] - 7s 13ms/step -
Epoch 15/15
500/500 [==============================] - 7s 15ms/step -
```

4. Average time for Test = 0s 6ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”

   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”

   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”

   Drop out Layer with rate = 0.5

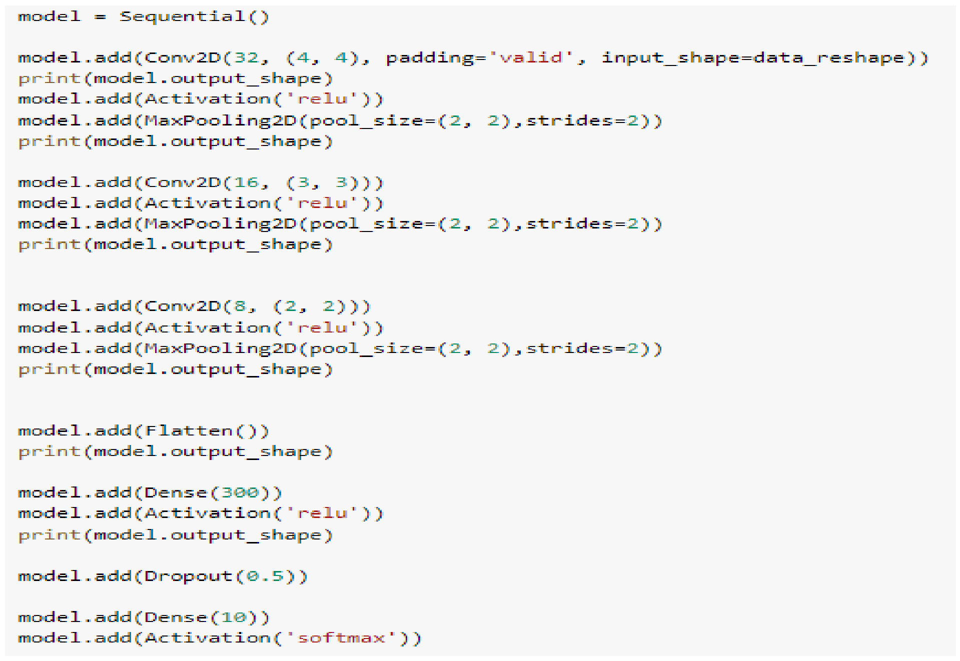
   FC_hidden1: Size=300, activation function= “Relu”

   FC_output: Size=10, activation function= “SoftMax”

6. Learning Rate=0.001
7. Optimizer: Adaptive Moment Estimation (Adam).

Model 24:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)

model.add(Dropout(0.5))

model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 95.39%

   For train:

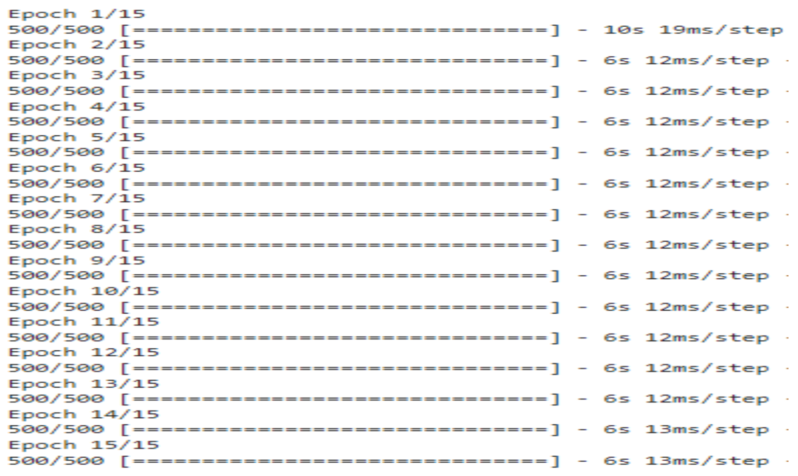
   - Epoch 1: accuracy: 0.4013
   - Epoch 2: accuracy: 0.7871
   - Epoch 3: accuracy: 0.9004
   - Epoch 4: accuracy: 0.9299
   - Epoch 5: accuracy: 0.9407

   For test:

   - Accuracy: 0.9540

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 10s 19ms/step
Epoch 2/15
500/500 [==============================] - 6s 12ms/step
Epoch 3/15
500/500 [==============================] - 6s 12ms/step
Epoch 4/15
500/500 [==============================] - 6s 12ms/step
Epoch 5/15
500/500 [==============================] - 6s 12ms/step
Epoch 6/15
500/500 [==============================] - 6s 12ms/step
Epoch 7/15
500/500 [==============================] - 6s 12ms/step
Epoch 8/15
500/500 [==============================] - 6s 12ms/step
Epoch 9/15
500/500 [==============================] - 6s 12ms/step
Epoch 10/15
500/500 [==============================] - 6s 12ms/step
Epoch 11/15
500/500 [==============================] - 6s 12ms/step
Epoch 12/15
500/500 [==============================] - 6s 12ms/step
Epoch 13/15
500/500 [==============================] - 6s 12ms/step
Epoch 14/15
500/500 [==============================] - 6s 13ms/step
Epoch 15/15
500/500 [==============================] - 6s 13ms/step
```

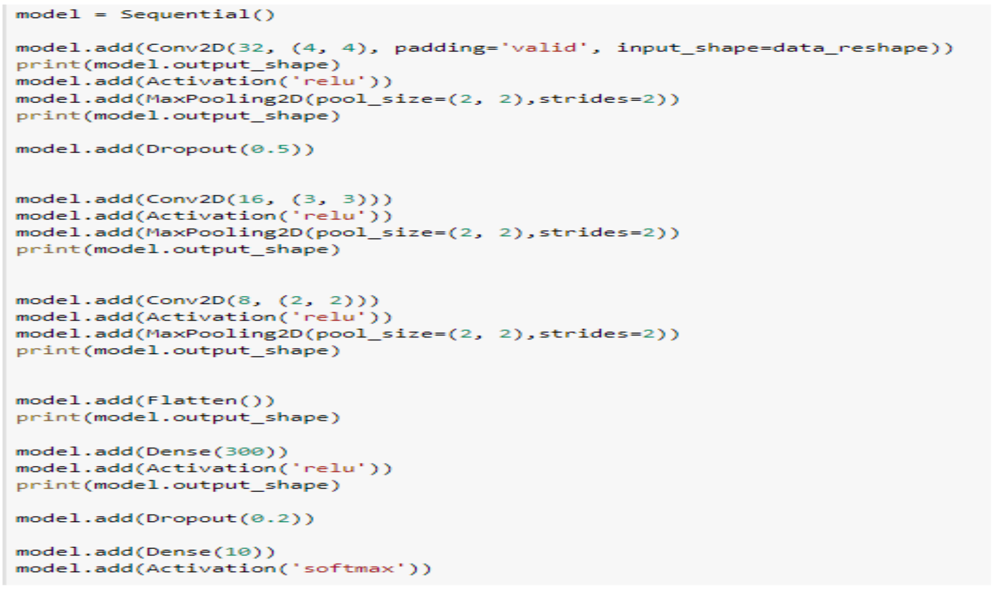
4. Average time for Test = 0s 5ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   Drop out Layer with rate = 0.5
   FC_output: Size=10, activation function= “SoftMax”
6. Learning Rate=0.001
7. Optimizer: Adaptive Moment Estimation (Adam).

____

Model 25:

- Epoch = 15
- Batch Size = 20



```
model = Sequential()

model.add(Conv2D(32, (4, 4), padding='valid', input_shape=data_reshape))
print(model.output_shape)
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)

model.add(Dropout(0.5))


model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Conv2D(8, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),strides=2))
print(model.output_shape)


model.add(Flatten())
print(model.output_shape)

model.add(Dense(300))
model.add(Activation('relu'))
print(model.output_shape)

model.add(Dropout(0.2))

model.add(Dense(10))
model.add(Activation('softmax'))
```

Testing:

1. Final Accuracy: 95.70%

   For train:

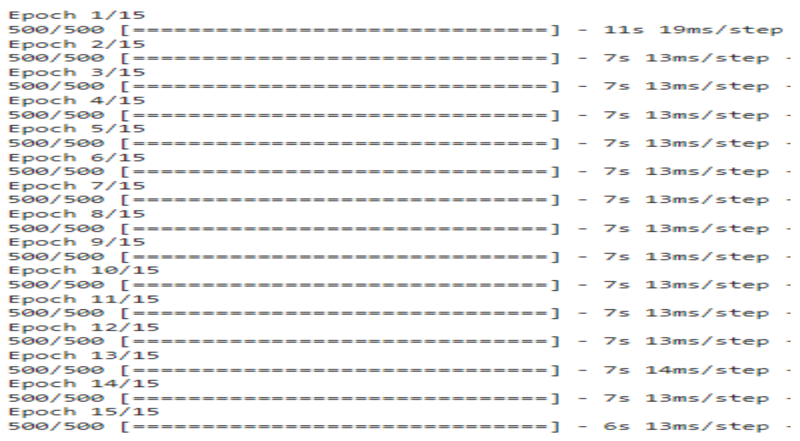
   - Epoch 1: accuracy: 0.4783
   - Epoch 2: accuracy: 0.7867
   - Epoch 3: accuracy: 0.8631
   - Epoch 4: accuracy: 0.8933
   - Epoch 5: accuracy: 0.9095

   For test:

   - Accuracy: 0.9580

2. Conv1 = $(4\ X\ 4\ X\ 1)32 + 32 = 544$
   Conv2 = $(3\ X\ 3\ X\ 32)\ 16 + 16 = 4624$
   Conv3 = $(2\ X\ 2\ X\ 16)\ 8 + 8 = 520$
   FC_hidden1= $(2\ X\ 2\ X\ 8)\ X\ 300 + 300 = \ 9900$
   FC_output = $300X\ 10 + 10 = 3010$
   Number of Parameters = 18598 parameters

3. Average time for Train:



```
Epoch 1/15
500/500 [==============================] - 11s 19ms/step
Epoch 2/15
500/500 [==============================] - 7s 13ms/step -
Epoch 3/15
500/500 [==============================] - 7s 13ms/step -
Epoch 4/15
500/500 [==============================] - 7s 13ms/step -
Epoch 5/15
500/500 [==============================] - 7s 13ms/step -
Epoch 6/15
500/500 [==============================] - 7s 13ms/step -
Epoch 7/15
500/500 [==============================] - 7s 13ms/step -
Epoch 8/15
500/500 [==============================] - 7s 13ms/step -
Epoch 9/15
500/500 [==============================] - 7s 13ms/step -
Epoch 10/15
500/500 [==============================] - 7s 13ms/step -
Epoch 11/15
500/500 [==============================] - 7s 13ms/step -
Epoch 12/15
500/500 [==============================] - 7s 13ms/step -
Epoch 13/15
500/500 [==============================] - 7s 14ms/step -
Epoch 14/15
500/500 [==============================] - 7s 13ms/step -
Epoch 15/15
500/500 [==============================] - 6s 13ms/step -
```

4. Average time for Test =
   0s 5ms/step

5. Conv1: Filters=32, kernel Size= 4 x 4, activation function= “Relu”
   Drop out Layer with rate = 0.5
   Conv2: Filters=16, kernel Size= 3 x 3, activation function= “Relu”
   Conv3: Filters=8, kernel Size= 2 x 2, activation function= “Relu”
   FC_hidden1: Size=300, activation function= “Relu”
   Drop out Layer with rate = 0.2
   FC_Output: Size=10, activation function= “SoftMax”
6. Learning Rate=0.001
7. Optimizer: Adaptive Moment Estimation (Adam).

Best Case: Model 21 where adding drop out =0.5 Layer after first Conv with Accuracy 96.8%

- **Epochs= 15**
- **Batch Size=20**
- **Learning Rate=0.001**
- **Optimizer is Adaptive Moment Estimation (Adam).**
- **Conv1: Filters=32, kernel Size= 4 x 4, activation function= "Relu"**
- **Drop out Layer with rate = 0.5**
- **Conv2: Filters=16, kernel Size= 3 x 3, activation function= "Relu"**
- **Conv3: Filters=8, kernel Size= 2 x 2, activation function= "Relu"**
- **FC_hidden1: Size=300, activation function= "Relu"**
- **FC_output: Size=10, activation function= "SoftMax"**

With Final Accuracy = 96.8%