1 Introduction

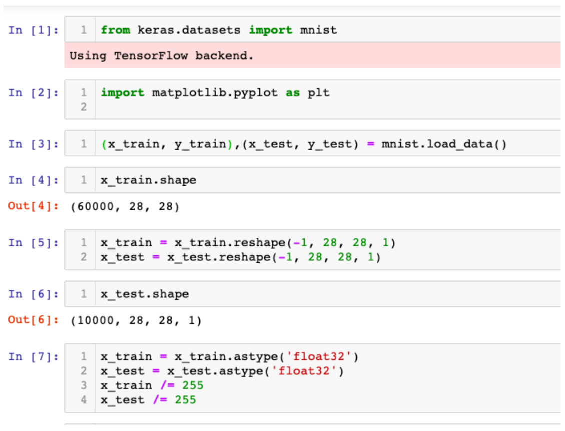
Implementation of CNN on MNIST dataset.

2 Dataset Description

The MNIST database of handwritten digits, has a training set of 60,000 examples, and a test set of 10,000 examples.
The dataset contain gray-scale images of hand-drawn digits, from zero through nine. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total.

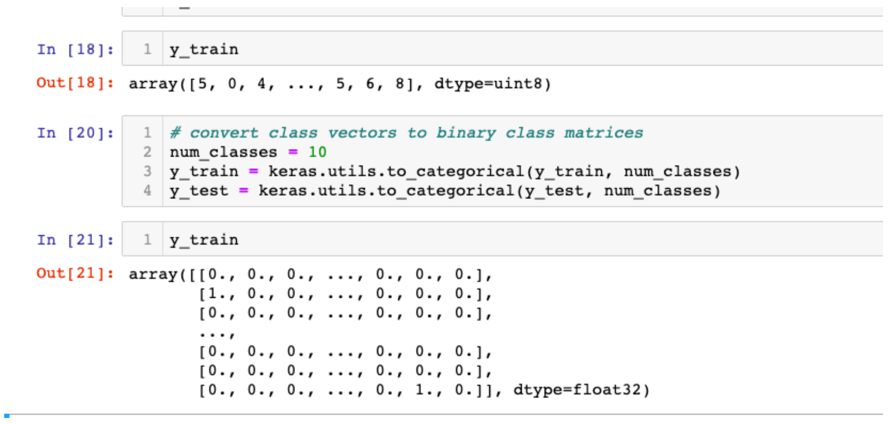
3 Pre-Processing of Data

As the pixel intensities are currently between the range of 0 and 255, Normalize the features, using broadcasting.(Figure-1)
Convert labels from a class vector to binary One Hot Encoded.(Figure-2)



```
In [1]:  from keras.datasets import mnist
         Using TensorFlow backend.

In [2]:  import matplotlib.pyplot as plt


In [3]:  (x_train, y_train),(x_test, y_test) = mnist.load_data()

In [4]:  x_train.shape
Out[4]:  (60000, 28, 28)

In [5]:  x_train = x_train.reshape(-1, 28, 28, 1)
         x_test = x_test.reshape(-1, 28, 28, 1)

In [6]:  x_test.shape
Out[6]:  (10000, 28, 28, 1)

In [7]:  x_train = x_train.astype('float32')
         x_test = x_test.astype('float32')
         x_train /= 255
         x_test /= 255
```

Figure 1: Pre-Processing

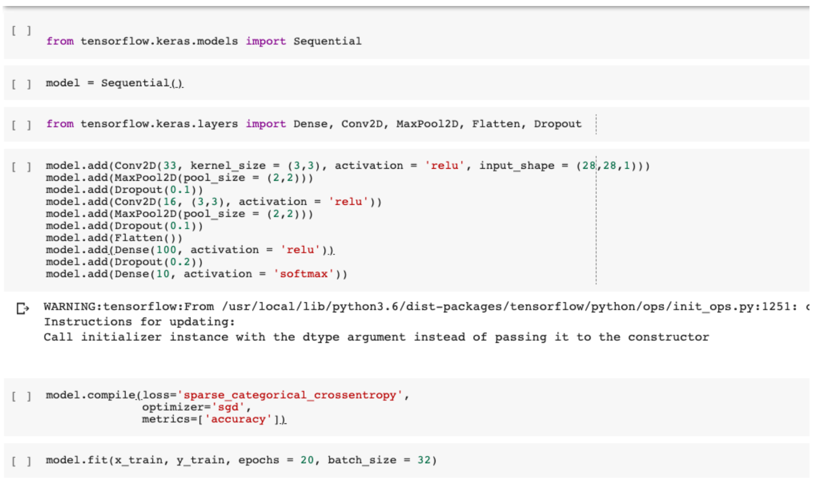


```
In [18]:  y_train
Out[18]:  array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)

In [20]:  # convert class vectors to binary class matrices
          num_classes = 10
          y_train = keras.utils.to_categorical(y_train, num_classes)
          y_test = keras.utils.to_categorical(y_test, num_classes)

In [21]:  y_train
Out[21]:  array([[0., 0., 0., ..., 0., 0., 0.],
                 [1., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 1., 0.]], dtype=float32)
```

Figure 2: Pre-Processing

4 CNN Model

A 4 layer Neural Network with MaxPool2D, Dropout and Flatten layer is trained using a batch size of 32 and with 20 epochs. (Figure-3)

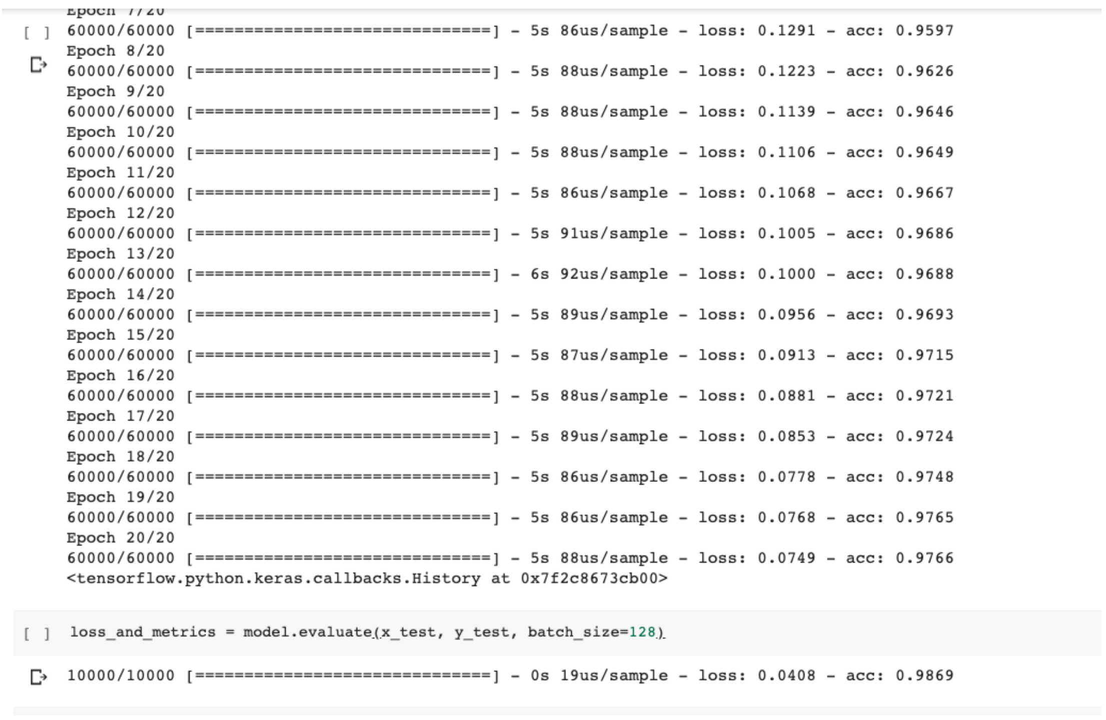


```
[ ]
    from tensorflow.keras.models import Sequential

[ ] model = Sequential()

[ ] from tensorflow.keras.layers import Dense, Conv2D, MaxPool2D, Flatten, Dropout

[ ] model.add(Conv2D(33, kernel_size = (3,3), activation = 'relu', input_shape = (28,28,1)))
    model.add(MaxPool2D(pool_size = (2,2)))
    model.add(Dropout(0.1))
    model.add(Conv2D(16, (3,3), activation = 'relu'))
    model.add(MaxPool2D(pool_size = (2,2)))
    model.add(Dropout(0.1))
    model.add(Flatten())
    model.add(Dense(100, activation = 'relu'))
    model.add(Dropout(0.2))
    model.add(Dense(10, activation = 'softmax'))

    WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/init_ops.py:1251: c
    Instructions for updating:
    Call initializer instance with the dtype argument instead of passing it to the constructor

[ ] model.compile(loss='sparse_categorical_crossentropy',
                  optimizer='sgd',
                  metrics=['accuracy'])

[ ] model.fit(x_train, y_train, epochs = 20, batch_size = 32)
```

Figure 3: CNN Model

5 Result

Accuracy(Training): 97.66
Accuracy(Test): 98.69
Loss: 0.040767

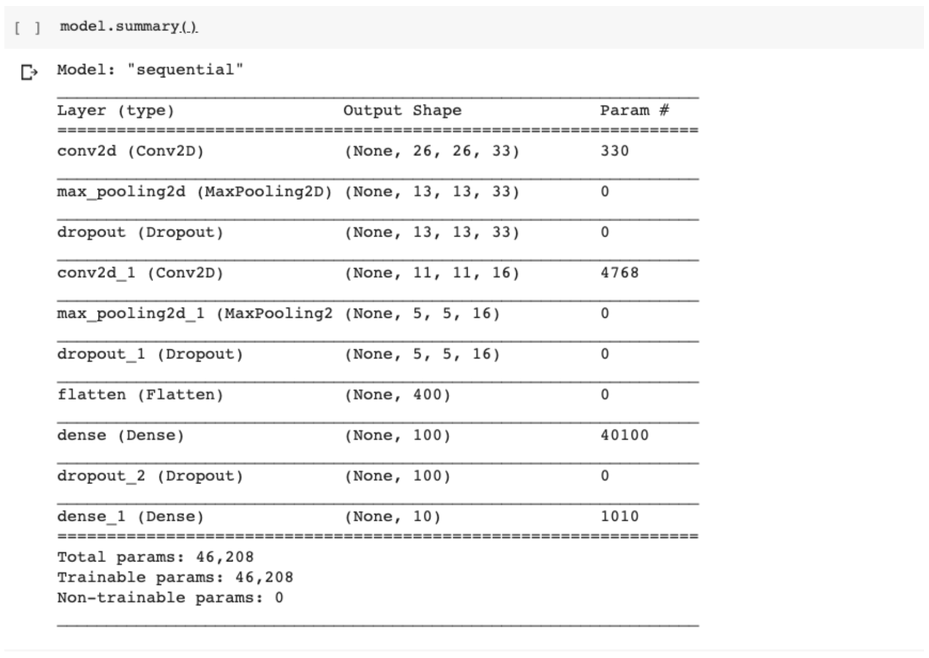


```
Epoch 7/20
60000/60000 [==============================] - 5s 86us/sample - loss: 0.1291 - acc: 0.9597
Epoch 8/20
60000/60000 [==============================] - 5s 88us/sample - loss: 0.1223 - acc: 0.9626
Epoch 9/20
60000/60000 [==============================] - 5s 88us/sample - loss: 0.1139 - acc: 0.9646
Epoch 10/20
60000/60000 [==============================] - 5s 88us/sample - loss: 0.1106 - acc: 0.9649
Epoch 11/20
60000/60000 [==============================] - 5s 86us/sample - loss: 0.1068 - acc: 0.9667
Epoch 12/20
60000/60000 [==============================] - 5s 91us/sample - loss: 0.1005 - acc: 0.9686
Epoch 13/20
60000/60000 [==============================] - 6s 92us/sample - loss: 0.1000 - acc: 0.9688
Epoch 14/20
60000/60000 [==============================] - 5s 89us/sample - loss: 0.0956 - acc: 0.9693
Epoch 15/20
60000/60000 [==============================] - 5s 87us/sample - loss: 0.0913 - acc: 0.9715
Epoch 16/20
60000/60000 [==============================] - 5s 88us/sample - loss: 0.0881 - acc: 0.9721
Epoch 17/20
60000/60000 [==============================] - 5s 89us/sample - loss: 0.0853 - acc: 0.9724
Epoch 18/20
60000/60000 [==============================] - 5s 86us/sample - loss: 0.0778 - acc: 0.9748
Epoch 19/20
60000/60000 [==============================] - 5s 86us/sample - loss: 0.0768 - acc: 0.9765
Epoch 20/20
60000/60000 [==============================] - 5s 88us/sample - loss: 0.0749 - acc: 0.9766
<tensorflow.python.keras.callbacks.History at 0x7f2c8673cb00>
```

```
[ ] loss_and_metrics = model.evaluate(x_test, y_test, batch_size=128)
```

```
10000/10000 [==============================] - 0s 19us/sample - loss: 0.0408 - acc: 0.9869
```

Figure 4: Training Model



```
[ ] model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 33)        330
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 13, 13, 33)        0
_________________________________________________________________
dropout (Dropout)            (None, 13, 13, 33)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 11, 11, 16)        4768
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 16)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 5, 5, 16)          0
_________________________________________________________________
flatten (Flatten)            (None, 400)               0
_________________________________________________________________
dense (Dense)                (None, 100)               40100
_________________________________________________________________
dropout_2 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1010
=================================================================
Total params: 46,208
Trainable params: 46,208
Non-trainable params: 0
_________________________________________________________________
```

Figure 5: Model Summary