Build a DNN using Keras with `RELU` and `ADAM`

Load tensorflow

```
from __future__ import absolute_import, division, print_function, unicode_literals
import tensorflow as tf
from matplotlib import pyplot as plt
import numpy as np
```

```
import tensorflow.compat.v1 as tf1
tf1.disable_v2_behavior()
```

Collect Fashion mnist data from tf.keras.datasets

```
tf.keras.datasets.fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
32768/29515 [================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
26427392/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-l
8192/5148 [===============================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-i
4423680/4422102 [==============================] - 0s 0us/step
((array([[[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, ..., 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],
          ...,
          [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, ..., 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]],

         ...,

         [[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, ..., 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],
          ...,
          [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, ..., 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]]], dtype=uint8),
```

```
 array([9, 0, 0, ..., 3, 0, 5], dtype=uint8)),
(array([[[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, ..., 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],
         ...,
         [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, ..., 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]],

        ...,

        [[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, ..., 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],
         ...,
         [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, ..., 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]]], dtype=uint8),
```

bold text#### Change train and test labels into one-hot vectors

```
import numpy as np
from keras.utils import np_utils
```

```
from keras.preprocessing.image import ImageDataGenerator


x_train = x_train.reshape(x_train.shape[0], 28, 28, 1).astype('float32')
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1).astype('float32')


x_train.shape
```

```
(60000, 28, 28, 1)
```

```
x_test.shape
```

```
(10000, 28, 28, 1)
```

```
y_train = np_utils.to_categorical(y_train, 10)
y_test = np_utils.to_categorical(y_test, 10)


y_train.shape
```

```
(60000, 10)
```

```
y_train[0].shape
```

```
(10,)
```

```
y_train[0]
```

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 1.], dtype=float32)
```

```
y_test.shape
```

```
(10000, 10)
```

```
y_test[0].shape
```

```
(10,)
```

```
y_test[0]
```

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 1.], dtype=float32)
```

```
x_train.shape
```

```
(60000, 28, 28, 1)
```

Build the Graph

Initialize model, reshape & normalize data

```
import keras
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout, Flatten, Reshape
from keras.layers import Convolution2D, MaxPooling2D
```

```
x_train=x_train.astype("float32")
```

```
x_test=x_test.astype("float32")
```

```
x_train /= 200
x_test /= 200
```

```
x_train[0]
```

```
array([[[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.   ],
        [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.   ],
 [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.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.005],
 [0.   ],
 [0.   ],
 [0.065],
 [0.365],
 [0.   ],
 [0.   ],
 [0.005],
 [0.02 ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.005],
 [0.005],
 [0.   ]],
```

```
[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.015],
 [0.   ],
 [0.18 ],
 [0.68 ],
 [0.635],
 [0.31 ],
 [0.27 ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.005],
 [0.015],
 [0.02 ],
 [0.   ],
 [0.   ],
 [0.015]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.03 ],
 [0.   ],
 [0.51 ],
 [1.02 ],
 [0.88 ],
 [0.67 ],
 [0.72 ],
 [0.615],
 [0.115],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.06 ],
 [0.05 ],
 [0.   ]]
```

```
 [0.   ]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.775],
 [1.18 ],
 [1.035],
 [0.89 ],
 [0.535],
 [0.78 ],
 [0.805],
 [0.545],
 [0.32 ],
 [0.115],
 [0.385],
 [0.65 ],
 [0.36 ],
 [0.075]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.005],
 [0.   ],
 [0.345],
 [1.035],
 [1.115],
 [1.09 ],
 [1.08 ],
 [1.08 ],
 [0.815],
 [0.635],
 [0.605],
 [0.61 ],
 [0.73 ],
 [0.705],
 [0.44 ],
 [0.86 ],
```

```
       [0.33 ]],

      [[0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.005],
       [0.005],
       [0.005],
       [0.   ],
       [1.   ],
       [1.16 ],
       [1.16 ],
       [1.165],
       [1.145],
       [1.115],
       [1.115],
       [1.075],
       [1.065],
       [0.82 ],
       [0.635],
       [0.615],
       [0.98 ],
       [1.145],
       [0.   ]],

      [[0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.915],
       [1.125],
       [1.08 ],
       [1.115],
       [1.14 ],
       [1.175],
       [1.135],
       [1.12 ],
       [1.11 ],
       [1.12 ],
       [1.105],
       [1.115],
       [1.225],
       [0.865],
```

```
  [0.   ]],

 [[0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.965],
  [1.14 ],
  [1.09 ],
  [1.065],
  [0.99 ],
  [0.9  ],
  [1.06 ],
  [1.05 ],
  [1.055],
  [1.065],
  [1.115],
  [1.1  ],
  [1.215],
  [1.01 ],
  [0.   ]],

 [[0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.   ],
  [0.005],
  [0.015],
  [0.   ],
  [0.06 ],
  [1.095],
  [1.1  ],
  [1.06 ],
  [1.09 ],
  [0.96 ],
  [0.845],
  [1.135],
  [1.04 ],
  [1.09 ],
  [1.12 ],
  [1.06 ],
  [1.13 ],
  [0.985],
  [1.045],
```

```
 [1.045],
 [0.26 ]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.03 ],
 [0.   ],
 [0.495],
 [1.22 ],
 [1.11 ],
 [1.1  ],
 [1.09 ],
 [1.015],
 [0.99 ],
 [1.105],
 [1.075],
 [1.065],
 [1.11 ],
 [1.1  ],
 [1.225],
 [0.595],
 [0.835],
 [0.28 ]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.02 ],
 [0.   ],
 [0.   ],
 [0.275],
 [1.18 ],
 [1.14 ],
 [1.15 ],
 [1.14 ],
 [1.2  ],
 [1.16 ],
 [1.065],
 [1.09 ],
 [1.115],
 [1.17 ],
 [1.085],
 [1.085],
 [1.045],
```

```
       [0.46 ],
       [0.   ]],

      [[0.   ],
       [0.   ],
       [0.005],
       [0.02 ],
       [0.03 ],
       [0.035],
       [0.01 ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [1.185],
       [1.13 ],
       [1.085],
       [1.115],
       [1.11 ],
       [1.095],
       [1.11 ],
       [1.105],
       [1.08 ],
       [1.115],
       [1.145],
       [1.075],
       [1.09 ],
       [1.275],
       [0.385],
       [0.   ]],

      [[0.   ],
       [0.015],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.31 ],
       [0.725],
       [1.02 ],
       [1.14 ],
       [1.035],
       [1.065],
       [1.105],
       [1.09 ],
       [1.04 ],
       [1.055],
       [1.09 ],
       [1.12 ],
       [1.115],
       [1.095],
       [1.075],
       [1.12 ],
       [1.22 ],
```

```
       [0.795],
       [0.   ]],

      [[0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.09 ],
       [0.22 ],
       [0.41 ],
       [0.535],
       [0.945],
       [1.14 ],
       [1.1  ],
       [1.11 ],
       [1.085],
       [1.13 ],
       [1.   ],
       [1.025],
       [1.055],
       [1.15 ],
       [1.12 ],
       [1.17 ],
       [0.88 ],
       [0.94 ],
       [1.25 ],
       [1.24 ],
       [1.165],
       [1.19 ],
       [1.075],
       [0.   ]],

      [[0.   ],
       [0.285],
       [0.935],
       [1.04 ],
       [1.12 ],
       [1.105],
       [1.12 ],
       [1.04 ],
       [1.02 ],
       [1.07 ],
       [1.04 ],
       [1.045],
       [1.   ],
       [0.795],
       [1.225],
       [0.965],
       [1.03 ],
       [1.115],
       [1.275],
       [1.275],
       [1.105],
       [1.17 ],
       [1.105],
       [1.055],
       [1.1  ],
       [1.16 ],
```

```
 [1.16 ],
 [1.23 ],
 [0.   ]],

[[0.015],
 [1.01 ],
 [1.14 ],
 [1.12 ],
 [1.105],
 [1.055],
 [1.055],
 [1.07 ],
 [1.025],
 [1.025],
 [1.025],
 [1.1  ],
 [1.2  ],
 [0.4  ],
 [0.75 ],
 [1.275],
 [1.145],
 [1.105],
 [0.94 ],
 [0.77 ],
 [0.955],
 [1.05 ],
 [1.02 ],
 [1.045],
 [1.11 ],
 [1.14 ],
 [1.125],
 [0.   ]],

[[0.49 ],
 [1.165],
 [0.99 ],
 [1.05 ],
 [1.11 ],
 [1.145],
 [1.145],
 [1.17 ],
 [1.245],
 [1.1  ],
 [0.97 ],
 [1.075],
 [1.085],
 [1.205],
 [0.325],
 [0.365],
 [0.53 ],
 [0.585],
 [0.84 ],
 [1.095],
 [1.105],
 [1.075],
 [1.085],
 [1.115],
 [1.115],
```

```
       [1.12 ],
       [1.145],
       [0.145]],

      [[0.375],
       [1.02 ],
       [1.06 ],
       [1.02 ],
       [0.965],
       [1.025],
       [1.055],
       [1.125],
       [1.08 ],
       [0.925],
       [0.985],
       [1.03 ],
       [0.99 ],
       [1.065],
       [1.2  ],
       [0.975],
       [1.135],
       [1.225],
       [1.195],
       [1.115],
       [1.09 ],
       [1.06 ],
       [1.045],
       [1.11 ],
       [1.1  ],
       [1.105],
       [1.15 ],
       [0.335]],

      [[0.24 ],
       [1.015],
       [0.915],
       [0.97 ],
       [1.065],
       [0.985],
       [0.925],
       [0.95 ],
       [0.97 ],
       [0.96 ],
       [1.01 ],
       [1.07 ],
       [1.095],
       [1.105],
       [1.1  ],
       [1.18 ],
       [1.125],
       [1.08 ],
       [0.995],
       [1.03 ],
       [0.93 ],
       [0.905],
       [0.885],
       [0.86 ],
       [0.905],
```

```
       [1.025],
       [1.03 ],
       [0.575]],

      [[0.   ],
       [0.61 ],
       [1.095],
       [0.965],
       [0.895],
       [0.855],
       [0.915],
       [0.98 ],
       [1.02 ],
       [1.05 ],
       [1.065],
       [1.035],
       [1.055],
       [1.05 ],
       [1.   ],
       [0.98 ],
       [0.97 ],
       [0.955],
       [0.975],
       [0.955],
       [0.99 ],
       [0.96 ],
       [0.88 ],
       [0.78 ],
       [0.835],
       [0.885],
       [1.05 ],
       [0.46 ]],

      [[0.   ],
       [0.   ],
       [0.37 ],
       [0.945],
       [1.06 ],
       [0.955],
       [0.875],
       [0.86 ],
       [0.875],
       [0.905],
       [0.925],
       [0.94 ],
       [0.945],
       [0.94 ],
       [0.965],
       [0.99 ],
       [1.02 ],
       [1.045],
       [1.05 ],
       [1.05 ],
       [1.055],
       [0.94 ],
       [0.94 ],
       [0.97 ],
       [0.96 ],
```

```
 [0.96 ],
 [1.08 ],
 [0.85 ],
 [0.   ]],

[[0.01 ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.33 ],
 [1.   ],
 [1.11 ],
 [1.185],
 [1.195],
 [1.21 ],
 [1.23 ],
 [1.215],
 [1.22 ],
 [1.105],
 [1.1  ],
 [0.965],
 [0.955],
 [0.895],
 [0.91 ],
 [0.91 ],
 [0.905],
 [0.88 ],
 [0.83 ],
 [0.84 ],
 [0.495],
 [0.29 ],
 [0.   ],
 [0.   ]],

[[0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.   ],
 [0.2  ],
 [0.305],
 [0.22 ],
 [0.36 ],
 [0.205],
 [0.175],
 [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.   ],
       [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.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
       [0.   ],
```

Add two fully connected layers with 200 and 100 neurons respectively with `relu` activatio

```
TRAIN = False
BATCH_SIZE = 32
EPOCHS = 10



model1 = Sequential()

# 1st Conv Layer
model1.add(Convolution2D(32, 3, 3, input_shape=(28, 28, 1)))
model1.add(Activation('relu'))

# 2nd Conv Layer
model1.add(Convolution2D(32, 3, 3))
model1.add(Activation('relu'))

# Fully Connected Layer
model1.add(Flatten())
model1.add(Dense(200))
model1.add(Activation('relu'))

# Prediction Layer
model1.add(Dense(10))
model1.add(Activation('softmax'))

# Loss and Optimizer
model1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Store Training Results
early_stopping = keras.callbacks.EarlyStopping(monitor='val_acc', patience=0.25, verbose=1, m
callback_list = [early_stopping]

# Train the model2
model1.fit(x_train, y_train, batch_size=BATCH_SIZE, nb_epoch=EPOCHS,
              validation_data=(x_test, y_test), callbacks=callback_list)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: UserWarning: Update your
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: UserWarning: Update your

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:29: UserWarning: The `nb_ep
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 216s 4ms/step - loss: 0.3614 - acc: 0.867
Epoch 2/10
60000/60000 [==============================] - 215s 4ms/step - loss: 0.2156 - acc: 0.919
Epoch 3/10
60000/60000 [==============================] - 214s 4ms/step - loss: 0.1476 - acc: 0.945
Epoch 4/10
60000/60000 [==============================] - 216s 4ms/step - loss: 0.0937 - acc: 0.965
Epoch 00004: early stopping
<keras.callbacks.History at 0x7fdaa65e78d0>
```

Add the output layer with a fully connected layer with 10 neurons with `softmax` ac `categorical_crossentropy` loss and `adam` optimizer and train the network. And, r

```
BATCH_SIZE = 32
EPOCHS = 10



# Define Model
model2 = Sequential()

# 1st Conv Layer
model2.add(Convolution2D(32, 3, 3, input_shape=(28, 28, 1)))
model2.add(Activation('relu'))

# 2nd Conv Layer
model2.add(Convolution2D(32, 3, 3))
model2.add(Activation('relu'))

# Max Pooling
model2.add(MaxPooling2D(pool_size=(2,2)))

# Dropout
model2.add(Dropout(0.25))

# Fully Connected Layer
model2.add(Flatten())
model2.add(Dense(200))
model2.add(Activation('relu'))

# Prediction Layer
model2.add(Dense(10))
```

```
model2.add(Activation('softmax'))

# Loss and Optimizer
model2.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Store Training Results
early_stopping = keras.callbacks.EarlyStopping(monitor='val_acc', patience=0.25, verbose=1, m
callback_list = [early_stopping]

# Train the model
model2.fit(x_train, y_train, batch_size=BATCH_SIZE, nb_epoch=EPOCHS,
              validation_data=(x_test, y_test), callbacks=callback_list)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: UserWarning: Update your
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: UserWarning: Update your

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:35: UserWarning: The `nb_ep
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 130s 2ms/step - loss: 0.3835 - acc: 0.861
Epoch 2/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.2482 - acc: 0.907
Epoch 3/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.2003 - acc: 0.925
Epoch 4/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.1647 - acc: 0.938
Epoch 5/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.1350 - acc: 0.948
Epoch 6/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.1128 - acc: 0.958
Epoch 7/10
60000/60000 [==============================] - 129s 2ms/step - loss: 0.0931 - acc: 0.964
Epoch 00007: early stopping
<keras.callbacks.History at 0x7fdaa6682470>
```