INFORMATION ABOUT DATA:

THE MNIST DATA SET IS IN BUILT IN TESNOR FLOW AKI KERAS

THE DATASET CONTAINS OF HAND WRITTEN DIGITS IN TH E FORM OF IMAGES

OBJECTIVE:

TO TRY DIFFERENT CONVOLUTIONAL NEURAL NETS

TO IMPLEMENT DIFFERENT KERNELS

TO TRY DIFFERENT NUMBER OF LAYERS IN A CONVOLUT IONAL NEURAN NETWORK

TO TRY MAX POOLING

TO INTRODUCE DROPOUT IN THE NETWORK

IMPORTING THE REQUIRED LIBRARIES AND DATA

In [1]:

```
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as k
from __future__ import print_function
import keras

(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Using TensorFlow backend.
```

In [2]:

```
batch_size = 1000
epochs = 10
img_rows, img_cols = 28, 28
if k.image_data_format() == 'channel_first':
  x_train = x_train.reshape(x_train.shape[0],1,img_rows,img_cols)
  x_test = x_test.reshape(x_test.shape[0],1,img_rows, img_cols)
  input_shape = (1, img_rows, img_cols)
```

```
else:
  x_train = x_train.reshape(x_train.shape[0],img_rows, img_cols, 1)
  x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols,1)
  input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
y_train = keras.utils.to_categorical(y_train, num_classes = 10)
y_test = keras.utils.to_categorical(y_test, num_classes = 10)
print("Train data:",x_train.shape)
print("Train labels:",y_train.shape)
print("Test data:",x_test.shape)
print("Test labels:",y_test.shape)
```

```
Train data: (60000, 28, 28, 1)
Train labels: (60000, 10)
Test data: (10000, 28, 28, 1)
Test labels: (10000, 10)
```

MODEL 1: 3 LAYER CONVOLUTIONAL NETWORK WITH KERNEL OF SI
E(3,3)

In [11]:

```
  model = Sequential()
  model.add(Conv2D(32, kernel_size=(3,3), activation = 'relu', input_shape =
input_shape))
  model.add(MaxPooling2D(pool_size=(2,2)))
  model.add(Dropout(0.5))
  model.add(Conv2D(64, kernel_size=(3,3), activation = 'relu'))
  model.add(MaxPooling2D(pool_size=(2,2)))
  model.add(Dropout(0.4))
  model.add(Conv2D(128, kernel_size=(3,3), activation= 'relu'))
  model.add(MaxPooling2D(pool_size=(2,2)))
  model.add(Dropout(0.3))
  model.add(Flatten())
  model.add(Dense(256, activation='relu'))
  model.add(Dense(10, activation='softmax'))
  model.compile(loss = keras.losses.categorical_crossentropy, optimizer = k
eras.optimizers.adam(), metrics = ['accuracy'] )
  model.fit(x_train,y_train, batch_size = batch_size, epochs = epochs,
validation_data = (x_test, y_test))
  score = model.evaluate(x_test, y_test)
  print("Test loss:", score[0])
  print("Test accuracy:", score[1])
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/20
60000/60000 [==============================] - 69s 1ms/step - loss: 1.3549
- acc: 0.5417 - val_loss: 0.4705 - val_acc: 0.8658
Epoch 2/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.5090
- acc: 0.8395 - val_loss: 0.2090 - val_acc: 0.9374
Epoch 3/20
```

```
60000/60000 [==============================] - 69s 1ms/step - loss: 0.3247
- acc: 0.9004 - val_loss: 0.1503 - val_acc: 0.9541
Epoch 4/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.2565
- acc: 0.9204 - val_loss: 0.1208 - val_acc: 0.9648
Epoch 5/20
20000/60000 [=========>....................] - ETA: 44s - loss: 0.2317 - ac
c: 0.928560000/60000 [==============================] - 69s 1ms/step - loss
: 0.2210 - acc: 0.9323 - val_loss: 0.0998 - val_acc: 0.9706
Epoch 6/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1943
- acc: 0.9404 - val_loss: 0.0881 - val_acc: 0.9738
Epoch 7/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1765
- acc: 0.9456 - val_loss: 0.0792 - val_acc: 0.9763
Epoch 8/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1617
- acc: 0.9513 - val_loss: 0.0729 - val_acc: 0.9776
Epoch 9/20
48000/60000 [=======================>......] - ETA: 13s - loss: 0.1537 - ac
c: 0.953260000/60000 [==============================] - 69s 1ms/step - loss
: 0.1520 - acc: 0.9535 - val_loss: 0.0686 - val_acc: 0.9797
Epoch 10/20
60000/60000 [==============================] - 68s 1ms/step - loss: 0.1416
- acc: 0.9565 - val_loss: 0.0664 - val_acc: 0.9807
Epoch 11/20
60000/60000 [==============================] - 68s 1ms/step - loss: 0.1339
- acc: 0.9594 - val_loss: 0.0623 - val_acc: 0.9813
Epoch 12/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1278
- acc: 0.9612 - val_loss: 0.0575 - val_acc: 0.9833
Epoch 13/20
51000/60000 [========================>.....] - ETA: 9s - loss: 0.1218 - acc
: 0.9617 60000/60000 [==============================] - 68s 1ms/step - loss
: 0.1216 - acc: 0.9621 - val_loss: 0.0582 - val_acc: 0.9817
Epoch 14/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1154
- acc: 0.9651 - val_loss: 0.0538 - val_acc: 0.9838
Epoch 15/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1098
- acc: 0.9664 - val_loss: 0.0531 - val_acc: 0.9848
Epoch 16/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.1063
- acc: 0.9680 - val_loss: 0.0515 - val_acc: 0.9844
Epoch 17/20
51000/60000 [========================>.....] - ETA: 9s - loss: 0.1002 - acc
: 0.9690 60000/60000 [==============================] - 68s 1ms/step - loss
: 0.1028 - acc: 0.9682 - val_loss: 0.0527 - val_acc: 0.9844
Epoch 18/20
60000/60000 [==============================] - 68s 1ms/step - loss: 0.0991
- acc: 0.9701 - val_loss: 0.0502 - val_acc: 0.9849
Epoch 19/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.0937
- acc: 0.9715 - val_loss: 0.0495 - val_acc: 0.9854
Epoch 20/20
60000/60000 [==============================] - 69s 1ms/step - loss: 0.0944
- acc: 0.9704 - val_loss: 0.0454 - val_acc: 0.9858
 8288/10000 [=======================>......] - ETA: 0s10000/10000 [========
==========================] - 4s 374us/step
Test loss: 0.045419275725237095
```

```
Test accuracy: 0.9858
```

MODEL 2 : 5 LAYER CONVOLUTIONAL NETWORK WITH KERNEL OF SIZE(5,5)

In [4]:

```
  model = Sequential()
  model.add(Conv2D(128, kernel_size=(5,5), activation= 'relu', input_shape =
input_shape))
  model.add(Dropout(0.5))
  model.add(Conv2D(64, kernel_size=(5,5), activation = 'relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(32, kernel_size=(5,5), activation = 'relu'))
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.5))
  model.add(Conv2D(64, kernel_size=(5,5), activation = 'relu'))
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.4))
  model.add(Conv2D(128, kernel_size=(5,5), activation= 'relu'))
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.3))
  model.add(Flatten())
  model.add(Dense(256, activation='relu'))
  model.add(Dense(10, activation='softmax'))
  model.compile(loss = keras.losses.categorical_crossentropy, optimizer = k
eras.optimizers.adam(), metrics = ['accuracy'] )
  model.fit(x_train,y_train, batch_size = batch_size, epochs = epochs,
validation_data = (x_test, y_test))
  score = model.evaluate(x_test, y_test)
  print("Test loss:", score[0])
  print("Test accuracy:", score[1])
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/20
60000/60000 [==============================] - 1428s 24ms/step - loss: 0.64
59 - acc: 0.7807 - val_loss: 0.1120 - val_acc: 0.9671
Epoch 2/20
60000/60000 [==============================] - 1447s 24ms/step - loss: 0.09
97 - acc: 0.9697 - val_loss: 0.0542 - val_acc: 0.9823
Epoch 3/20
60000/60000 [==============================] - 1415s 24ms/step - loss: 0.06
27 - acc: 0.9805 - val_loss: 0.0395 - val_acc: 0.9876
Epoch 4/20
60000/60000 [==============================] - 1441s 24ms/step - loss: 0.05
12 - acc: 0.9838 - val_loss: 0.0340 - val_acc: 0.9900
Epoch 5/20
17000/60000 [=======>......................] - ETA: 16:28 - loss: 0.0429 -
acc: 0.986060000/60000 [==============================] - 1436s 24ms/step -
loss: 0.0435 - acc: 0.9862 - val_loss: 0.0314 - val_acc: 0.9908
Epoch 6/20
60000/60000 [==============================] - 1440s 24ms/step - loss: 0.03
82 - acc: 0.9881 - val_loss: 0.0288 - val_acc: 0.9919
Epoch 7/20
60000/60000 [==============================] - 1434s 24ms/step - loss: 0.03
42 - acc: 0.9889 - val_loss: 0.0275 - val_acc: 0.9916
Epoch 8/20
```

```
60000/60000 [==============================] - 1437s 24ms/step - loss: 0.03
00 - acc: 0.9903 - val_loss: 0.0250 - val_acc: 0.9921
Epoch 9/20
44000/60000 [====================>.........] - ETA: 6:08 - loss: 0.0263 - a
cc: 0.991460000/60000 [==============================] - 1442s 24ms/step -
loss: 0.0260 - acc: 0.9917 - val_loss: 0.0266 - val_acc: 0.9921
Epoch 10/20
60000/60000 [==============================] - 1441s 24ms/step - loss: 0.02
58 - acc: 0.9919 - val_loss: 0.0263 - val_acc: 0.9917
Epoch 11/20
60000/60000 [==============================] - 1434s 24ms/step - loss: 0.02
31 - acc: 0.9927 - val_loss: 0.0267 - val_acc: 0.9921
Epoch 12/20
60000/60000 [==============================] - 1439s 24ms/step - loss: 0.02
04 - acc: 0.9934 - val_loss: 0.0256 - val_acc: 0.9917
Epoch 13/20
47000/60000 [======================>.......] - ETA: 4:59 - loss: 0.0190 - a
cc: 0.994160000/60000 [==============================] - 1440s 24ms/step -
loss: 0.0194 - acc: 0.9938 - val_loss: 0.0198 - val_acc: 0.9932
Epoch 14/20
60000/60000 [==============================] - 1432s 24ms/step - loss: 0.01
73 - acc: 0.9938 - val_loss: 0.0240 - val_acc: 0.9934
Epoch 15/20
60000/60000 [==============================] - 1434s 24ms/step - loss: 0.01
69 - acc: 0.9944 - val_loss: 0.0212 - val_acc: 0.9934
Epoch 16/20
60000/60000 [==============================] - 1435s 24ms/step - loss: 0.01
56 - acc: 0.9948 - val_loss: 0.0226 - val_acc: 0.9941
Epoch 17/20
47000/60000 [======================>.......] - ETA: 4:57 - loss: 0.0162 - a
cc: 0.994760000/60000 [==============================] - 1436s 24ms/step -
loss: 0.0163 - acc: 0.9944 - val_loss: 0.0224 - val_acc: 0.9936
Epoch 18/20
60000/60000 [==============================] - 1429s 24ms/step - loss: 0.01
58 - acc: 0.9949 - val_loss: 0.0237 - val_acc: 0.9939
Epoch 19/20
60000/60000 [==============================] - 1426s 24ms/step - loss: 0.01
36 - acc: 0.9954 - val_loss: 0.0203 - val_acc: 0.9937
Epoch 20/20
60000/60000 [==============================] - 1434s 24ms/step - loss: 0.01
21 - acc: 0.9958 - val_loss: 0.0259 - val_acc: 0.9925
 1600/10000 [===>..........................] - ETA: 53s10000/10000 [=======
==========================] - 63s 6ms/step
Test loss: 0.025934723713708807
Test accuracy: 0.9925
```

MODEL 3 : 7 LAYER CONVOLUTIONAL NETWORK WITH DIFFERENT KERNEL SIZE IN EACH LAYER

In [3]:

```
  model = Sequential()
  model.add(Conv2D(128, kernel_size=(4,4), activation= 'relu', input_shape =
input_shape))
  model.add(Dropout(0.5))
  model.add(Conv2D(64, kernel_size=(3,3), activation = 'relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(32, kernel_size=(2,2), activation = 'relu'))
```

```
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.5))
  model.add(Conv2D(64, kernel_size=(5,5), activation = 'relu'))
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.4))
  model.add(Conv2D(128, kernel_size=(5,5), activation= 'relu'))
  model.add(MaxPooling2D(pool_size=(1,1)))
  model.add(Dropout(0.3))
  model.add(Conv2D(64, kernel_size=(2,2), activation= 'relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(32, kernel_size=(3,3), activation='relu'))
  model.add(Dropout(0.5))
  model.add(Flatten())
  model.add(Dense(256, activation='relu'))
  model.add(Dense(10, activation='softmax'))
  model.compile(loss = keras.losses.categorical_crossentropy, optimizer = k
eras.optimizers.adam(), metrics = ['accuracy'] )
  model.fit(x_train,y_train, batch_size = batch_size, epochs = epochs,
validation_data = (x_test, y_test))
  score = model.evaluate(x_test, y_test)
  print("Test loss:", score[0])
  print("Test accuracy:", score[1])
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 1226s 20ms/step - loss: 0.78
40 - acc: 0.7442 - val_loss: 0.1810 - val_acc: 0.9447
Epoch 2/10
60000/60000 [==============================] - 1227s 20ms/step - loss: 0.16
01 - acc: 0.9503 - val_loss: 0.0664 - val_acc: 0.9781
Epoch 3/10
60000/60000 [==============================] - 1222s 20ms/step - loss: 0.09
05 - acc: 0.9726 - val_loss: 0.0493 - val_acc: 0.9839
Epoch 4/10
60000/60000 [==============================] - 1216s 20ms/step - loss: 0.07
04 - acc: 0.9787 - val_loss: 0.0421 - val_acc: 0.9862
Epoch 5/10
17000/60000 [=======>......................] - ETA: 13:59 - loss: 0.0622 -
acc: 0.981260000/60000 [==============================] - 1219s 20ms/step -
loss: 0.0571 - acc: 0.9826 - val_loss: 0.0410 - val_acc: 0.9878
Epoch 6/10
60000/60000 [==============================] - 1220s 20ms/step - loss: 0.05
37 - acc: 0.9835 - val_loss: 0.0334 - val_acc: 0.9893
Epoch 7/10
60000/60000 [==============================] - 1220s 20ms/step - loss: 0.04
68 - acc: 0.9854 - val_loss: 0.0285 - val_acc: 0.9910
Epoch 8/10
60000/60000 [==============================] - 1222s 20ms/step - loss: 0.04
04 - acc: 0.9872 - val_loss: 0.0315 - val_acc: 0.9900
Epoch 9/10
44000/60000 [=====================>........] - ETA: 5:12 - loss: 0.0373 - a
cc: 0.988360000/60000 [==============================] - 1222s 20ms/step -
loss: 0.0372 - acc: 0.9881 - val_loss: 0.0296 - val_acc: 0.9904
Epoch 10/10
60000/60000 [==============================] - 1222s 20ms/step - loss: 0.03
56 - acc: 0.9890 - val_loss: 0.0255 - val_acc: 0.9921
 8320/10000 [=======================>......] - ETA: 8s10000/10000 [========
======================] - 51s 5ms/step
Test loss: 0.025468569558369927
```

```
Test accuracy: 0.9921


            CONCLUSION:

                        EPOCHS = 20

                        MODEL 1 : 3 LAYER CONVOLUTIONAL NETWORK WITH KEI
   NEL SIZE (3,3)

                                              TEST LOSS : 0.04541      A
   CURACY : 98.58

                        MODEL 2 : 5 LAYER CONVOLUTIONAL NETWORK WITH KEI
   NEL SIZE (5,5)

                                             TEST LOSS : 0.02593      AC
   URACY : 99.25

                         EPOCHS = 10

                         MODEL 3 : 7 LAYER CONVOLUTIONAL NETWORK WITH DI
   FFERENT KERNEL SIZE IN EACH LAYER

                                             TEST LOSS : 0.02546      AC
   URACY : 99.21
```