1st ARCHITECTURE

KERNEL (3,3)

In [0]:
```
from __future__ import print_function
import keras
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

batch_size = 128
num_classes = 10
epochs = 12

img_rows , img_cols = 28,28
```

```
Using TensorFlow backend.
```

In [0]:
```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 1s 0us/step
```

In [0]:
```
if K.image_data_format() == 'channels_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)
```

In [0]:
```
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
```

In [0]:
```
print ('x_train shape :' , x_train.shape)
print (x_train.shape[0] , 'train samples')
print (x_test.shape[0] , 'test samples')


y_train = keras.utils.to_categorical(y_train , num_classes)
y_test = keras.utils.to_categorical(y_test , num_classes)

model = Sequential()

model.add(Conv2D (32, kernel_size=(3, 3), activation='relu', input_shap
e = input_shape))

model.add(Conv2D(64, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))

model.add(Conv2D(256, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))
model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.5))



model.add(Dense(num_classes, activation='softmax'))

model.compile(optimizer=keras.optimizers.Adadelta(),
              loss=keras.losses.categorical_crossentropy,
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=epochs, batch_size=batch_size , verb
ose = 1 , validation_data = (x_test , y_test))


score = model.evaluate(x_test, y_test, verbose=0)
print('loss=', score[0])
print('accuracy=', score[1])
```

```
x_train shape : (60000, 28, 28, 1)
60000 train samples
10000 test samples
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
60000/60000 [==============================] - 254s 4ms/step - loss: 1
4.0194 - acc: 0.1121 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 2/12
60000/60000 [==============================] - 256s 4ms/step - loss: 1
4.2927 - acc: 0.1131 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 3/12
60000/60000 [==============================] - 258s 4ms/step - loss: 1
4.2615 - acc: 0.1151 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 4/12
60000/60000 [==============================] - 252s 4ms/step - loss: 1
4.2814 - acc: 0.1138 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 5/12
60000/60000 [==============================] - 251s 4ms/step - loss: 1
```

```
4.2459 - acc: 0.1161 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 6/12
60000/60000 [==============================] - 258s 4ms/step - loss: 1
4.2620 - acc: 0.1151 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 7/12
60000/60000 [==============================] - 269s 4ms/step - loss: 1
4.2575 - acc: 0.1154 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 8/12
60000/60000 [==============================] - 269s 4ms/step - loss: 1
4.2618 - acc: 0.1152 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 9/12
60000/60000 [==============================] - 272s 5ms/step - loss: 1
4.2709 - acc: 0.1146 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 10/12
60000/60000 [==============================] - 271s 5ms/step - loss: 1
4.2605 - acc: 0.1153 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 11/12
60000/60000 [==============================] - 269s 4ms/step - loss: 1
4.2801 - acc: 0.1140 - val_loss: 14.2887 - val_acc: 0.1135
Epoch 12/12
60000/60000 [==============================] - 271s 5ms/step - loss: 1
4.2859 - acc: 0.1136 - val_loss: 14.2887 - val_acc: 0.1135
loss= 14.28869146270752
accuracy= 0.1135
```

2ND ARCHITECTURE

KERNEL (5,5)

In [0]:
```
from __future__ import print_function
import keras
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
```

```
batch_size = 128
num_classes = 10
epochs = 12

img_rows , img_cols = 28,28

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

if K.image_data_format() == 'channels_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

print ('x_train shape :' , x_train.shape)
print (x_train.shape[0] , 'train samples')
print (x_test.shape[0] , 'test samples')


y_train = keras.utils.to_categorical(y_train , num_classes)
y_test = keras.utils.to_categorical(y_test , num_classes)

model = Sequential()

model.add(Conv2D (32, kernel_size=(5, 5), activation='relu', input_shap
e = input_shape))

model.add(Conv2D(128, (5 , 5), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))



model.add(Conv2D(80, (5 , 5), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))





model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.5))


model.add(Dense(num_classes, activation='softmax'))

model.compile(optimizer=keras.optimizers.Adadelta(),
              loss=keras.losses.categorical_crossentropy,
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=epochs, batch_size=batch_size , verb
ose = 1 , validation_data = (x_test , y_test))


score = model.evaluate(x_test, y_test, verbose=0)
```

```
print('loss=', score[0])
print('accuracy=', score[1])
```

```
Using TensorFlow backend.
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 1s 0us/step
x_train shape : (60000, 28, 28, 1)
60000 train samples
10000 test samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/framework/op_def_library.py:263: colocate_with (from tensorfl
ow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/ba
ckend/tensorflow_backend.py:3445: calling dropout (from tensorflow.pyth
on.ops.nn_ops) with keep_prob is deprecated and will be removed in a fu
ture version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate =
1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.ma
th_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
60000/60000 [==============================] - 500s 8ms/step - loss: 0.
4158 - acc: 0.8722 - val_loss: 0.0935 - val_acc: 0.9739
Epoch 2/12
60000/60000 [==============================] - 496s 8ms/step - loss: 0.
1849 - acc: 0.9510 - val_loss: 0.0634 - val_acc: 0.9831
Epoch 3/12
60000/60000 [==============================] - 495s 8ms/step - loss: 0.
1392 - acc: 0.9631 - val_loss: 0.0510 - val_acc: 0.9844
Epoch 4/12
60000/60000 [==============================] - 495s 8ms/step - loss: 0.
```

```
1180 - acc: 0.9690 - val_loss: 0.0465 - val_acc: 0.9845
Epoch 5/12
60000/60000 [==============================] - 497s 8ms/step - loss: 0.
0976 - acc: 0.9737 - val_loss: 0.0402 - val_acc: 0.9883
Epoch 6/12
60000/60000 [==============================] - 496s 8ms/step - loss: 0.
0885 - acc: 0.9770 - val_loss: 0.0358 - val_acc: 0.9888
Epoch 7/12
60000/60000 [==============================] - 496s 8ms/step - loss: 0.
0790 - acc: 0.9791 - val_loss: 0.0417 - val_acc: 0.9889
Epoch 8/12
60000/60000 [==============================] - 498s 8ms/step - loss: 0.
0766 - acc: 0.9801 - val_loss: 0.0299 - val_acc: 0.9905
Epoch 9/12
60000/60000 [==============================] - 497s 8ms/step - loss: 0.
0689 - acc: 0.9829 - val_loss: 0.0272 - val_acc: 0.9913
Epoch 10/12
60000/60000 [==============================] - 497s 8ms/step - loss: 0.
0701 - acc: 0.9823 - val_loss: 0.0285 - val_acc: 0.9920
Epoch 11/12
60000/60000 [==============================] - 498s 8ms/step - loss: 0.
0663 - acc: 0.9836 - val_loss: 0.0327 - val_acc: 0.9923
Epoch 12/12
60000/60000 [==============================] - 497s 8ms/step - loss: 0.
0641 - acc: 0.9846 - val_loss: 0.0321 - val_acc: 0.9920
loss= 0.03214728945528709
accuracy= 0.992
```

3RD ARCHITECTURE

KERNEL (7,7)

In [0]:
```
from __future__ import print_function
import keras
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

batch_size = 128
num_classes = 10
epochs = 12

img_rows , img_cols = 28,28

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

if K.image_data_format() == 'channels_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

print ('x_train shape :' , x_train.shape)
print (x_train.shape[0] , 'train samples')
print (x_test.shape[0] , 'test samples')


y_train = keras.utils.to_categorical(y_train , num_classes)
y_test = keras.utils.to_categorical(y_test , num_classes)

model = Sequential()

model.add(Conv2D (32, kernel_size=(7, 7), activation='relu', input_shap
e = input_shape))
```

```
model.add(Conv2D(128, (7 , 7), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))




model.add(Conv2D(80, (7 , 7), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.25))






model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.5))



model.add(Dense(num_classes, activation='softmax'))

model.compile(optimizer=keras.optimizers.Adadelta(),
              loss=keras.losses.categorical_crossentropy,
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=epochs, batch_size=batch_size , verb
ose = 1 , validation_data = (x_test , y_test))
```

```
score = model.evaluate(x_test, y_test, verbose=0)
print('loss=', score[0])
print('accuracy=', score[1])
```

```
Using TensorFlow backend.
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 4s 0us/step
x_train shape : (60000, 28, 28, 1)
60000 train samples
10000 test samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/framework/op_def_library.py:263: colocate_with (from tensorfl
ow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/ba
ckend/tensorflow_backend.py:3445: calling dropout (from tensorflow.pyth
on.ops.nn_ops) with keep_prob is deprecated and will be removed in a fu
ture version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate =
1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.ma
th_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
60000/60000 [==============================] - 518s 9ms/step - loss: 2.
3511 - acc: 0.1103 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 2/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3104 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 3/12
60000/60000 [==============================] - 521s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
```

```
Epoch 4/12
60000/60000 [==============================] - 521s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 5/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 6/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3147 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 7/12
60000/60000 [==============================] - 517s 9ms/step - loss: 2.
3078 - acc: 0.1123 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 8/12
60000/60000 [==============================] - 515s 9ms/step - loss: 2.
3144 - acc: 0.1123 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 9/12
60000/60000 [==============================] - 517s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 10/12
60000/60000 [==============================] - 518s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 11/12
60000/60000 [==============================] - 520s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 12/12
33152/60000 [===============>..............] - ETA: 3:44 - loss: 2.3010
- acc: 0.1144
```

```
Using TensorFlow backend.
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 4s 0us/step
x_train shape : (60000, 28, 28, 1)
60000 train samples
10000 test samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/framework/op_def_library.py:263: colocate_with (from tensorfl
ow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
```

```
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/ba
ckend/tensorflow_backend.py:3445: calling dropout (from tensorflow.pyth
on.ops.nn_ops) with keep_prob is deprecated and will be removed in a fu
ture version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate =
1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorfl
ow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.ma
th_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
60000/60000 [==============================] - 518s 9ms/step - loss: 2.
3511 - acc: 0.1103 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 2/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3104 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 3/12
60000/60000 [==============================] - 521s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 4/12
60000/60000 [==============================] - 521s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 5/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 6/12
60000/60000 [==============================] - 522s 9ms/step - loss: 2.
3147 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 7/12
60000/60000 [==============================] - 517s 9ms/step - loss: 2.
3078 - acc: 0.1123 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 8/12
60000/60000 [==============================] - 515s 9ms/step - loss: 2.
3144 - acc: 0.1123 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 9/12
```

```
60000/60000 [==============================] - 517s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 10/12
60000/60000 [==============================] - 518s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3011 - val_acc: 0.1135
Epoch 11/12
60000/60000 [==============================] - 520s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
Epoch 12/12
60000/60000 [==============================] - 526s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
60000/60000 [==============================] - 526s 9ms/step - loss: 2.
3013 - acc: 0.1124 - val_loss: 2.3010 - val_acc: 0.1135
loss= 2.301046071624756
accuracy= 0.1135
loss= 2.301046071624756
accuracy= 0.1135
```