Annapoornima S

225229101

PDL Lab11. Exploration of Convolutional Neural Networks Design

In [7]:

```
from __future__ import print_function
import keras
from keras.datasets import cifar10
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.utils import to_categorical
import tensorflow as tf
from tensorflow.keras.optimizers import RMSprop
import matplotlib.pyplot as plt
%matplotlib inline
```

In [8]:

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

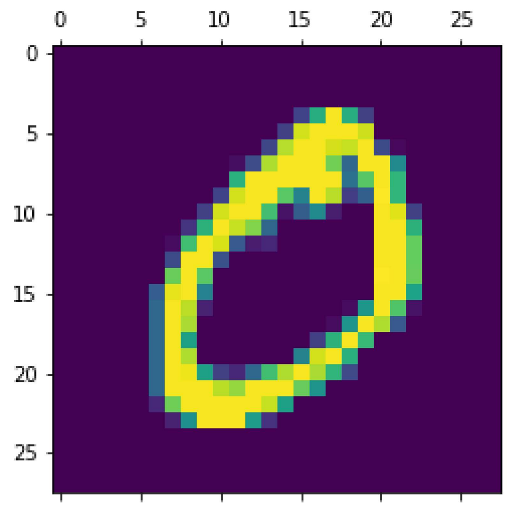
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz (https://storage.googleapis.com/tensorflow/tf-keras-da
tasets/mnist.npz)
11490434/11490434 [==============================] - 4s 0us/step
```

In [59]:

```
plt.matshow(x_train[1])
```

Out[59]:

```
<matplotlib.image.AxesImage at 0x21323c50430>
```



In [11]:

```
X_train = x_train.astype('float32')/255
X_test = x_test.astype('float32')/255
```

In [12]:

```
X_train.shape
```

Out[12]:

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

In [13]:

```
y_train.shape
```

Out[13]:

```
(60000,)
```

In [19]:

```
def mod(n):
    model = Sequential()
    model.add(Conv2D(filters=n, kernel_size=(3, 3), activation='relu', input_shape=(28,2
    model.add(Flatten())
    model.add(Dense(10,activation = 'softmax'))
    return model
```

In [20]:

```
model02=mod(4)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 11s 4ms/step - loss: 27.3046
- accuracy: 0.1039
Epoch 2/5
1875/1875 [==============================] - 7s 4ms/step - loss: 27.3046
- accuracy: 0.0996
Epoch 3/5
1875/1875 [==============================] - 7s 4ms/step - loss: 27.3046
- accuracy: 0.0999
Epoch 4/5
1875/1875 [==============================] - 7s 4ms/step - loss: 27.3046
- accuracy: 0.1000
Epoch 5/5
1875/1875 [==============================] - 7s 4ms/step - loss: 27.3046
- accuracy: 0.1014
```

Out[20]:

In [22]:

```
model02=mod(32)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 13s 7ms/step - loss: 27.3046
- accuracy: 0.0985
Epoch 2/5
1875/1875 [==============================] - 13s 7ms/step - loss: 27.3046
- accuracy: 0.1015
Epoch 3/5
1875/1875 [==============================] - 13s 7ms/step - loss: 27.3046
- accuracy: 0.1008
Epoch 4/5
1875/1875 [==============================] - 13s 7ms/step - loss: 27.3046
- accuracy: 0.1013
Epoch 5/5
1875/1875 [==============================] - 13s 7ms/step - loss: 27.3046
- accuracy: 0.1006
```

Out[22]:

```
<keras.src.callbacks.History at 0x2130161e6d0>
```

In [24]:

```
model02=mod(128)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 47s 25ms/step - loss: 27.304
6 - accuracy: 0.1019
Epoch 2/5
1875/1875 [==============================] - 47s 25ms/step - loss: 27.304
6 - accuracy: 0.0982
Epoch 3/5
1875/1875 [==============================] - 47s 25ms/step - loss: 27.304
6 - accuracy: 0.0952
Epoch 4/5
1875/1875 [==============================] - 49s 26ms/step - loss: 27.304
6 - accuracy: 0.0968
Epoch 5/5
1875/1875 [==============================] - 49s 26ms/step - loss: 27.304
6 - accuracy: 0.0989
```

Out[24]:

```
<keras.src.callbacks.History at 0x21301b57070>
```

In [25]:

```
def mod(n):
    model = Sequential()
    for i in range(n):
        model.add(Conv2D(filters=n, kernel_size=(3, 3), activation='relu', input_shape=(
        model.add(Flatten())
        model.add(Dense(10,activation = 'softmax'))
        return model
```

In [26]:

```
model02=mod(2)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5,batch_size=64)
```

```
Epoch 1/5
938/938 [==============================] - 5s 4ms/step - loss: 27.3045 -
accuracy: 0.1107
Epoch 2/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1088
Epoch 3/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1024
Epoch 4/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1041
Epoch 5/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1043
```

Out[26]:

```
<keras.src.callbacks.History at 0x21301bf1280>
```

In [27]:

```
model02=mod(3)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5,batch_size=64)
```

```
Epoch 1/5
938/938 [==============================] - 5s 4ms/step - loss: 27.3045 -
accuracy: 0.1003
Epoch 2/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.0951
Epoch 3/5
938/938 [==============================] - 5s 5ms/step - loss: 27.3045 -
accuracy: 0.0949
Epoch 4/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.0950
Epoch 5/5
938/938 [==============================] - 4s 5ms/step - loss: 27.3045 -
accuracy: 0.0958
```

Out[27]:

```
<keras.src.callbacks.History at 0x2130623ed30>
```

In [28]:

```
model02=mod(4)
model02.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model02.fit(X_train,y_train,epochs=5,batch_size=64)
```

```
Epoch 1/5
938/938 [==============================] - 5s 4ms/step - loss: 27.3045 -
accuracy: 0.0998
Epoch 2/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1016
Epoch 3/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1030
Epoch 4/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1023
Epoch 5/5
938/938 [==============================] - 4s 4ms/step - loss: 27.3045 -
accuracy: 0.1003
```

Out[28]:

```
<keras.src.callbacks.History at 0x213062dee50>
```

In [29]:

```
model01 = Sequential()
model01.add(Conv2D(filters=16, kernel_size=(5,5), activation='relu', input_shape=(28,28,
model01.add(Flatten())
model01.add(Dense(10,activation = 'softmax'))
model01.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model01.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.1008
Epoch 2/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.1034
Epoch 3/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.1024
Epoch 4/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.1015
Epoch 5/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1020
```

Out[29]:

```
<keras.src.callbacks.History at 0x21308d445e0>
```

In [30]:

```
model01 = Sequential()
model01.add(Conv2D(filters=16, kernel_size=(7,7), activation='relu', input_shape=(28,28,
model01.add(Flatten())
model01.add(Dense(10,activation = 'softmax'))
model01.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model01.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1067
Epoch 2/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.1022
Epoch 3/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1013
Epoch 4/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.0981
Epoch 5/5
1875/1875 [==============================] - 8s 4ms/step - loss: 27.3046
- accuracy: 0.0982
```

Out[30]:

In [32]:

```
def mod(n,act):
    model = Sequential()
    for i in range(n):
        model.add(Conv2D(filters=16, kernel_size=(3, 3), activation=act, input_shape=(28
        model.add(Flatten())
        model.add(Dense(10,activation = 'softmax'))
        return model
```

In [33]:

```
model01=mod(2,'tanh')
model01.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model01.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1024
Epoch 2/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1022
Epoch 3/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0996
Epoch 4/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1010
Epoch 5/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0997
```

Out[33]:

```
<keras.src.callbacks.History at 0x21309405880>
```

In [34]:

```
score = model01.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.251890182495117
Test accuracy: 0.10119999945163727
```

In [35]:

```
model01=mod(2,'relu')
model01.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model01.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1053
Epoch 2/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.0960
Epoch 3/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.0963
Epoch 4/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1015
Epoch 5/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0982
```

Out[35]:

```
<keras.src.callbacks.History at 0x2130af108b0>
```

In [36]:

```
score = model01.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.261587142944336
Test accuracy: 0.10769999772310257
```

In [40]:

```
model2 = Sequential()
model2.add(Conv2D(filters=16, kernel_size=(3, 3), activation='relu', input_shape=(28,28,
model2.add(Conv2D(filters=16, kernel_size=(5,5), activation='relu', input_shape=(28,28,1
model2.add(Flatten())
model2.add(Dense(10,activation = 'softmax'))
model2.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model2.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 48s 25ms/step - loss: 27.304
6 - accuracy: 0.0953
Epoch 2/5
1875/1875 [==============================] - 44s 23ms/step - loss: 27.304
6 - accuracy: 0.1022
Epoch 3/5
1875/1875 [==============================] - 43s 23ms/step - loss: 27.304
6 - accuracy: 0.1000
Epoch 4/5
1875/1875 [==============================] - 44s 23ms/step - loss: 27.304
6 - accuracy: 0.1002
Epoch 5/5
1875/1875 [==============================] - 44s 23ms/step - loss: 27.304
6 - accuracy: 0.0989
```

Out[40]:

```
<keras.src.callbacks.History at 0x2130b298fd0>
```

In [41]:

```
score = model2.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.256969451904297
Test accuracy: 0.13699999451637268
```

In [42]:

```
model3 = Sequential()
model3.add(Conv2D(filters=32, kernel_size=(3, 3),strides=(2,2), activation='relu', input
model3.add(Conv2D(filters=32, kernel_size=(5,5),strides=(2,2), activation='relu', input_
model3.add(Flatten())
model3.add(Dense(10,activation = 'softmax'))
model3.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model3.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 11s 5ms/step - loss: 27.3046
- accuracy: 0.0961
Epoch 2/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.0993
Epoch 3/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1016
Epoch 4/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1024
Epoch 5/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1011
```

Out[42]:

```
<keras.src.callbacks.History at 0x213093f0d30>
```

In [43]:

```
score = model3.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.256338119506836
Test accuracy: 0.0723000019788742
```

In [44]:

```
model4 = Sequential()
model4.add(Conv2D(filters=32, kernel_size=(3, 3),strides=(3,3), activation='relu', input
model4.add(Conv2D(filters=32, kernel_size=(5,5),strides=(3,3), activation='relu', input_
model4.add(Flatten())
model4.add(Dense(10,activation = 'softmax'))
model4.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model4.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 5s 2ms/step - loss: 27.3046
- accuracy: 0.0977
Epoch 2/5
1875/1875 [==============================] - 4s 2ms/step - loss: 27.3046
- accuracy: 0.1018
Epoch 3/5
1875/1875 [==============================] - 4s 2ms/step - loss: 27.3046
- accuracy: 0.1005
Epoch 4/5
1875/1875 [==============================] - 4s 2ms/step - loss: 27.3046
- accuracy: 0.1024
Epoch 5/5
1875/1875 [==============================] - 4s 2ms/step - loss: 27.3046
- accuracy: 0.1013
```

Out[44]:

```
<keras.src.callbacks.History at 0x2135077d3a0>
```

In [45]:

```
score = model4.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.256040573120117
Test accuracy: 0.07249999791383743
```

In [47]:

```
model5 = Sequential()
model5.add(Conv2D(filters=32, kernel_size=(5,5),strides=(2,2), activation='relu', input_
model5.add(Conv2D(filters=32, kernel_size=(5,5),strides=(2,2), activation='relu', input_
model5.add(Flatten())
model5.add(Dense(10,activation = 'softmax'))
model5.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model5.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 10s 5ms/step - loss: 27.3046
- accuracy: 0.1037
Epoch 2/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1012
Epoch 3/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1010
Epoch 4/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1009
Epoch 5/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1028
```

Out[47]:

```
<keras.src.callbacks.History at 0x21352aa9430>
```

In [48]:

```
score = model5.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.25255584716797
Test accuracy: 0.09390000253915787
```

In [49]:

```
model6 = Sequential()
model6.add(Conv2D(filters=32, kernel_size=(7,7),strides=(2,2), activation='relu', input_
model6.add(Conv2D(filters=32, kernel_size=(7,7),strides=(2,2), activation='relu', input_
model6.add(Flatten())
model6.add(Dense(10,activation = 'softmax'))
model6.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model6.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0981
Epoch 2/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0987
Epoch 3/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.0985
Epoch 4/5
1875/1875 [==============================] - 8s 5ms/step - loss: 27.3046
- accuracy: 0.1003
Epoch 5/5
1875/1875 [==============================] - 9s 5ms/step - loss: 27.3046
- accuracy: 0.1006
```

Out[49]:

```
<keras.src.callbacks.History at 0x21352a87820>
```

In [50]:

```
score = model6.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.250436782836914
Test accuracy: 0.046300001442432404
```

In [51]:

```
model7 = Sequential()
model7.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=(28,28,1
model7.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=(28,28,1
model7.add(MaxPooling2D(pool_size=(2,2)))
model7.add(Flatten())
model7.add(Dense(10,activation = 'softmax'))
model7.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model7.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 76s 40ms/step - loss: 27.304
5 - accuracy: 0.1011
Epoch 2/5
1875/1875 [==============================] - 75s 40ms/step - loss: 27.304
6 - accuracy: 0.0970
Epoch 3/5
1875/1875 [==============================] - 73s 39ms/step - loss: 27.304
6 - accuracy: 0.0994
Epoch 4/5
1875/1875 [==============================] - 75s 40ms/step - loss: 27.304
6 - accuracy: 0.0982
Epoch 5/5
1875/1875 [==============================] - 72s 39ms/step - loss: 27.304
6 - accuracy: 0.0992
```

Out[51]:

```
<keras.src.callbacks.History at 0x213013ab4c0>
```

In [52]:

```
score = model7.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.253005981445312
Test accuracy: 0.11999999731779099
```

In [53]:

```
model8 = Sequential()
model8.add(Conv2D(filters=32, kernel_size=(7,7), activation='relu', input_shape=(28,28,1
model8.add(Conv2D(filters=32, kernel_size=(7,7), activation='relu', input_shape=(28,28,1
model8.add(MaxPooling2D(pool_size=(2,2)))
model8.add(Flatten())
model8.add(Dense(10,activation = 'softmax'))
model8.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model8.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 58s 31ms/step - loss: 27.304
6 - accuracy: 0.0971
Epoch 2/5
1875/1875 [==============================] - 59s 31ms/step - loss: 27.304
6 - accuracy: 0.1018
Epoch 3/5
1875/1875 [==============================] - 59s 31ms/step - loss: 27.304
6 - accuracy: 0.1007
Epoch 4/5
1875/1875 [==============================] - 59s 31ms/step - loss: 27.304
6 - accuracy: 0.1032
Epoch 5/5
1875/1875 [==============================] - 60s 32ms/step - loss: 27.304
6 - accuracy: 0.1006
```

Out[53]:

```
<keras.src.callbacks.History at 0x213016198e0>
```

In [54]:

```
score = model8.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.250656127929688
Test accuracy: 0.05900000035762787
```

In [55]:

```
model9 = Sequential()
model9.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu',padding='same', input
model9.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu',padding='same', input
model9.add(Flatten())
model9.add(Dense(10,activation = 'softmax'))
model9.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])
model9.fit(X_train,y_train,epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 97s 52ms/step - loss: 27.304
6 - accuracy: 0.1022
Epoch 2/5
1875/1875 [==============================] - 101s 54ms/step - loss: 27.30
46 - accuracy: 0.1028
Epoch 3/5
1875/1875 [==============================] - 100s 53ms/step - loss: 27.30
46 - accuracy: 0.1033
Epoch 4/5
1875/1875 [==============================] - 98s 52ms/step - loss: 27.304
6 - accuracy: 0.0985
Epoch 5/5
1875/1875 [==============================] - 100s 53ms/step - loss: 27.30
46 - accuracy: 0.0996
```

Out[55]:

```
<keras.src.callbacks.History at 0x2130502f670>
```

In [56]:

```
score = model9.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 27.252906799316406
Test accuracy: 0.11360000073909760
```