Name: Muhammed Essam **ID:** 20190462

Name: Fatema Hesham **ID:** 20190373

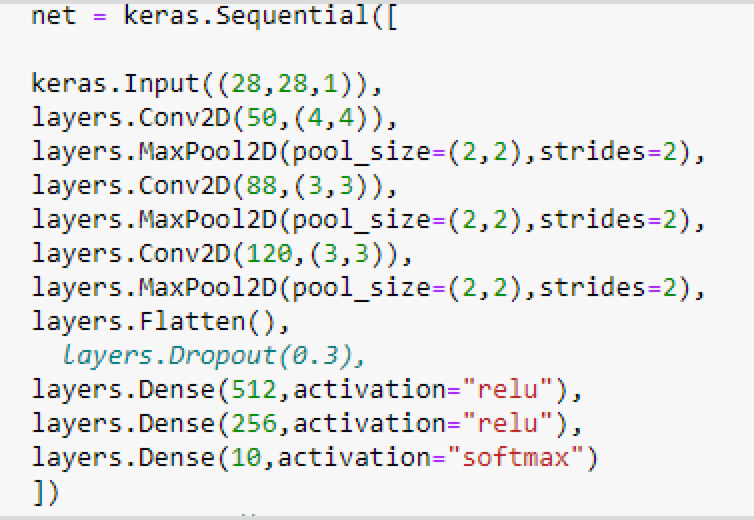
Mnist dataset:

The MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. It is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.

Sample of training = 15000, **sample of testing** = 10000

CNN:

1. Model first Structure:
   - Convolutions layers = 3 layers using max_pooling ((2,2),stride=2) and activation function after each layer ("Relu")
     - Num_channels on layer 1 = 50
     - Num_channels on layer 2 = 88
     - Num_channels on layer 3 = 120
   - Fully connected layers = 3 layers and output layer using activation function in hidden layer ("Relu") and ("Softmax") on output layer
     - Num_neurals on layer 1 = 120
     - Num_ neurals on layer 2 = 512
     - Num_ neurals on layer 3 = 256
     - Num_ neurals on output layer = 10
   - Optimizer = SGD, learning rate = 0.0009
   - Batch size = 32

- 

```
net = keras.Sequential([

keras.Input((28,28,1)),
layers.Conv2D(50,(4,4)),
layers.MaxPool2D(pool_size=(2,2),strides=2),
layers.Conv2D(88,(3,3)),
layers.MaxPool2D(pool_size=(2,2),strides=2),
layers.Conv2D(120,(3,3)),
layers.MaxPool2D(pool_size=(2,2),strides=2),
layers.Flatten(),
  layers.Dropout(0.3),
layers.Dense(512,activation="relu"),
layers.Dense(256,activation="relu"),
layers.Dense(10,activation="softmax")
])
```

<u>Find best epochs:</u>

Number of epochs we try is in range of [10:30] and best models in range of [10:20].

From

1. Model 1:
    - Number of epochs =14
    - Final accuracy on train = 99.9
    - Final accuracy on test = 96.4
    - Accuracy in first 5 epochs

```
o Epoch 1/14
o accuracy: 0.8056
o Epoch 2/14
o accuracy: 0.9302
o Epoch 3/14
o accuracy: 0.9533
o Epoch 4/14
o accuracy: 0.9673
o Epoch 5/14
o accuracy: 0.9772
```

    - Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

    - Avg_time in each epoch for training data = 21s

- Testing time = 4s

2. Model 2:
   - Number of epochs = 16
   - Final accuracy on train = 99.9
   - Final accuracy on test = 96.77
   - Accuracy in first 5 epochs

```
o  Epoch 1/16
o  accuracy: 0.8021
o  Epoch 2/16
o  accuracy: 0.9221
o  Epoch 3/16
o  accuracy: 0.9471
o  Epoch 4/16
o  accuracy: 0.9591
o  Epoch 5/16
o  accuracy: 0.9683
```

   - Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

   - Avg_time in each epoch for training data = 21s
   - Testing time = 4s

3. Model 3:
   - Number of epochs = 26
   - Final accuracy on train = 100.00%
   - Final accuracy on test = 96.81%
   - Accuracy in first 5 epochs

```
o  Epoch 1/26
o  accuracy: 0.8171
o  Epoch 2/26
o  accuracy: 0.9331
o  Epoch 3/26
o  accuracy: 0.9567
o  Epoch 4/26
o  accuracy: 0.9690
o  Epoch 5/26
o  accuracy: 0.9775
```

   - Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

- Avg_time in each epoch for training data = 19 s
- Testing time = 4s

4. Model 4:
   - Number of epochs = 19
   - Final accuracy on train = 100.00%
   - Final accuracy on test = 96.7%
   - Accuracy in first 5 epochs

```
o  Epoch 1/19
o  accuracy: 0.8055
o  Epoch 2/19
o  accuracy: 0.9276
o  Epoch 3/19
o  accuracy: 0.9535
o  Epoch 4/19
o  accuracy: 0.9637
o  Epoch 5/19
o  accuracy: 0.9735
```

- Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

- Avg_time in each epoch for training data = 21 s
- Testing time = 4s

Conclusion and observation:

- Accuracy increasing with the increase of number of epochs
- After 16 epochs time increasing and accuracy almost same but at 26 epochs accuracy will be 96.8% with time more than 16. At sample time with epoch = 320

  At sample time with 26 epochs = 500

So best model with 16 epochs

Changing in learning rate:

5. Model 5:
    - Learning rate = 0.002
    - Final accuracy on train = 100.00%
    - Final accuracy on test = 96.9%
    - Accuracy in first 5 epochs

```
o  1/16
o  accuracy: 0.8010
o  Epoch 2/16
o  accuracy: 0.9418
o  Epoch 3/16
o  accuracy: 0.9634
o  Epoch 4/16
o  accuracy: 0.9750
o  Epoch 5/16
o  accuracy: 0.9831
```

    - Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

    - Avg_time in each epoch for training data = 14s
    - Testing time = 4s

6. Model 6:
    - Learning rate = 0.001
    - Final accuracy on train = 100.00%
    - Final accuracy on test = 96.7%
    - Accuracy in first 5 epochs

```
o  Epoch 1/16
o  accuracy: 0.8145
o  Epoch 2/16
o  accuracy: 0.9335
o  Epoch 3/16
o  accuracy: 0.9573
o  Epoch 4/16
o  accuracy: 0.9699
o  Epoch 5/16
o  accuracy: 0.9785
```

    - Number of parameters:

```
o Total params: 331,548
o Trainable params: 331,548
```

- Avg_time in each epoch for training data = 15 s
- Testing time = 4s

7. Model 7:
   - Learning rate = 0.0005
   - Final accuracy on train = 99.7
   - Final accuracy on test = 95.7%
   - Accuracy in first 5 epochs

```
•   Epoch 1/16
•   accuracy: 0.7908
•   Epoch 2/16
•   accuracy: 0.9167
•   Epoch 3/16
•   accuracy: 0.9438
•   Epoch 4/16
•   accuracy: 0.9556
•   Epoch 5/16
•   accuracy: 0.9649
```

   - Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

   - Avg_time in each epoch for training data = 17s
   - Testing time = 4s

8. Model 8:
   - Learning rate = 0.0007
   - Final accuracy on train = 99.9%
   - Final accuracy on test = 96.2%
   - Accuracy in first 5 epochs

```
•   Epoch 1/16
•   accuracy: 0.7893
•   Epoch 2/16
•   accuracy: 0.9202
•   Epoch 3/16
•   accuracy: 0.9446
•   Epoch 4/16
•   accuracy: 0.9603
•   Epoch 5/16
•  accuracy: 0.9691
```

- Number of parameters:

```
o  Total params: 331,548
o Trainable params: 331,548
```

- Avg_time in each epoch for training data = 16s
- Testing time = 4s

Conclusion and observation:

- Best accuracy and best avg_time at learning rate = 0.002
- Total time = 242

So best model with 16 epochs and learning rate =0.002

Changing CNN kernel's size:

9. Model 9:

- At first layer changing to (2,2)
- At second layer changing to (3,3)
- At second layer changing to (2,2)
-
- Final accuracy on train = 100.00%
- Final accuracy on test = 97.67
- Accuracy in first 5 epochs

```
o  Epoch 1/16
o  accuracy: 0.8449
o  Epoch 2/16
o  accuracy: 0.9533
o  Epoch 3/16
o  accuracy: 0.9722
o  Epoch 4/16
o  accuracy: 0.9841
o  Epoch 5/16
o  accuracy: 0.9889
```

- Number of parameters:

```
o Total params: 395,866
o Trainable params: 395,866
```

- Avg_time in each epoch for training data = 21s

- Testing time = 6s

Changing in CNN number of channels and number of neurons:

10. Model 10:

- Number of channels at first layer = 128
- Number of channels at second layer = 88
- Number of channels at third layer = 120
-
- Final accuracy on train = 100.00%
- Final accuracy on test = 97.6%
- Accuracy in first 5 epochs

```
•  Epoch 1/16
•  accuracy: 0.8143
•  Epoch 2/16
•  accuracy: 0.9446
•  Epoch 3/16
•  accuracy: 0.9628
•  Epoch 4/16
•  accuracy: 0.9760
•  Epoch 5/16
•  accuracy: 0.9841
```

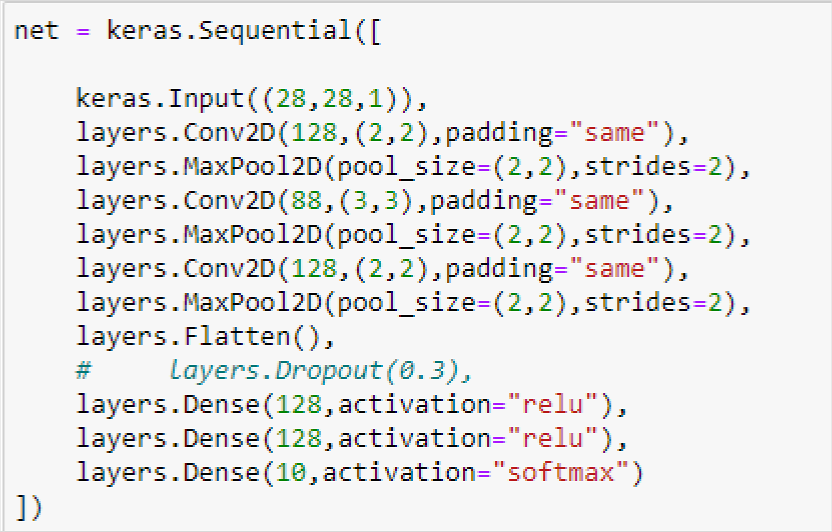
- Number of parameters:

```
o Total params: 267,682
o Trainable params: 267,682
```

- Avg_time in each epoch for training data = 41s
- Testing time = 6s

11.Model 11:

Begin with this structure



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="relu"),
    layers.Dense(128,activation="relu"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 100.00%
- Final accuracy on test = 97.3%
- Accuracy in first 5 epochs

```
o  Epoch 1/16
o  accuracy: 0.8189
o  Epoch 2/16
o  accuracy: 0.9435
o  Epoch 3/16
o  accuracy: 0.9635
o  Epoch 4/16
o  accuracy: 0.9721
o  Epoch 5/16
o  accuracy: 0.9815
```

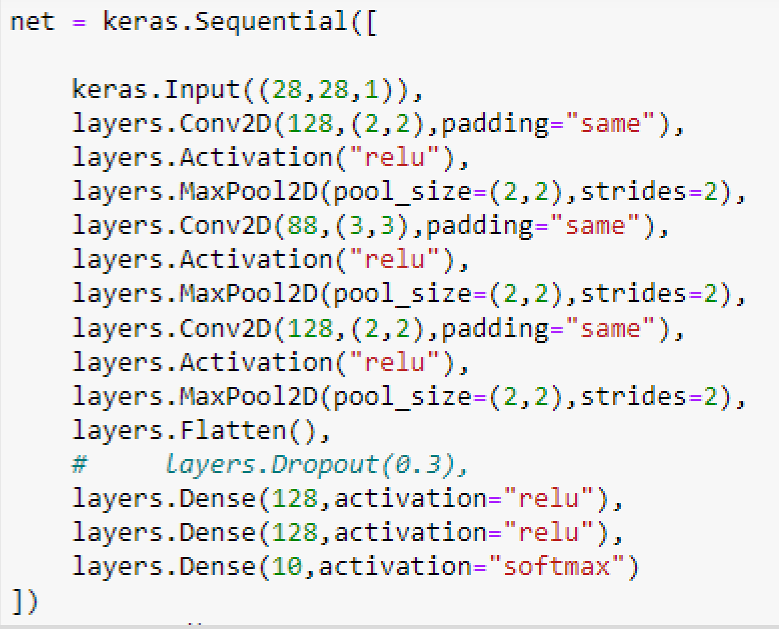
- Number of parameters:

```
o Total params: 230,754
o Trainable params: 230,754
```

- Avg_time in each epoch for training data = 43s
- Testing time = 9s

Try to set an activation function after each conv. layer:

12.Model 12:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="relu"),
    layers.Dense(128,activation="relu"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 100.00%
- Final accuracy on test = 98.17%
- Accuracy in first 5 epochs

```
o  Epoch 1/16
o  accuracy: 0.8616
o  Epoch 2/16
o  accuracy: 0.9518
o  Epoch 3/16
o  accuracy: 0.9685
o  Epoch 4/16
o  accuracy: 0.9767
o  Epoch 5/16
o  accuracy: 0.9830
```

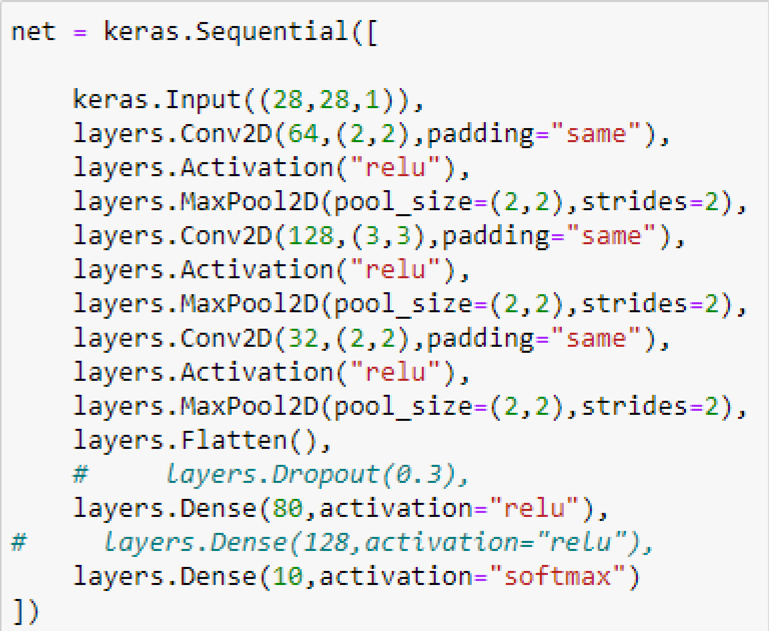
- Number of parameters:

```
o Total params: 312,674
o Trainable params: 312,674
```

- Avg_time in each epoch for training data = 50s
- Testing time = 12s

Changing on the number of layer, number of channels and number of neurons:

13.Model 13:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(64,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(3,3),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(32,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(80,activation="relu"),
#     layers.Dense(128,activation="relu"),
    layers.Dense(10,activation="softmax")
])
```

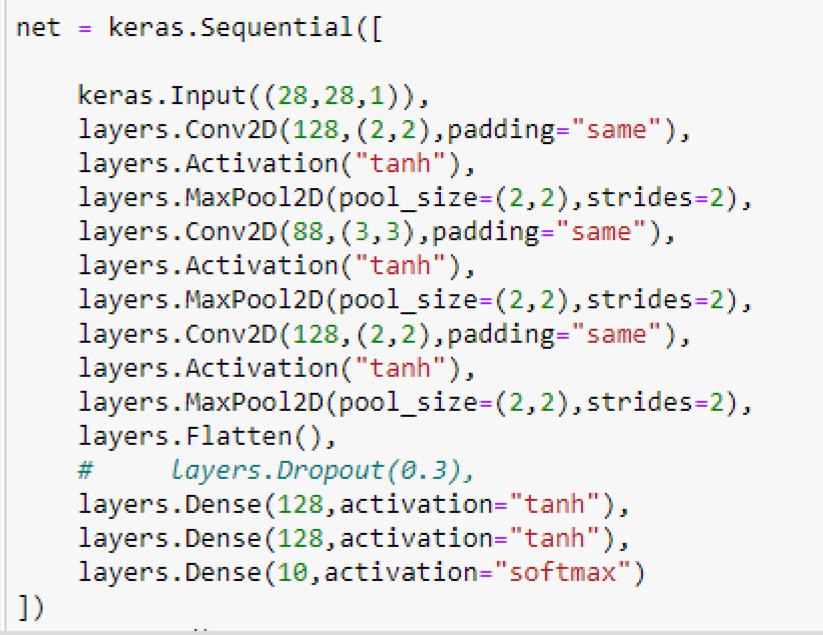
- Final accuracy on train = 100.00%
- Final accuracy on test = 97.79%
- Accuracy in first 5 epochs
  - Epoch 1/16
  - accuracy: 0.7871
  - Epoch 2/16
  - accuracy: 0.9416
  - Epoch 3/16
  - accuracy: 0.9592
  - Epoch 4/16
  - accuracy: 0.9681
  - Epoch 5/16
  - accuracy: 0.9718
- Number of parameters:
  - Total params: 114,522
  - Trainable params: 114,522
- Avg_time in each epoch for training data = 36s
- Testing time = 6s

Conclusion and observation:

- Best is model 12

Try best activation function:

14.Model 14:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="tanh"),
    layers.Dense(128,activation="tanh"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 100.00%
- Final accuracy on test = 97.2%
- Accuracy in first 5 epochs

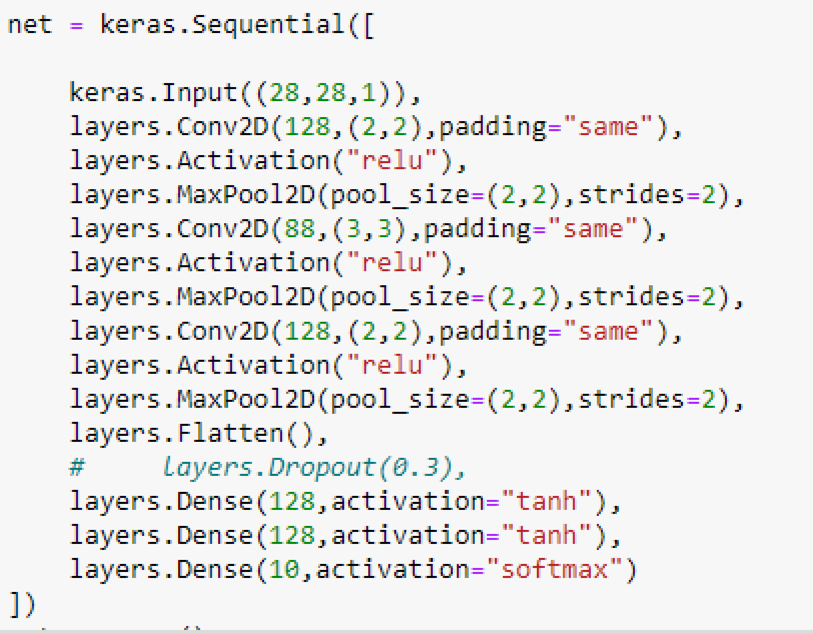
```
o  Epoch 1/16
o  accuracy: 0.6603
o  Epoch 2/16
o  accuracy: 0.8749
o  Epoch 3/16
o  accuracy: 0.9177
o  Epoch 4/16
o  accuracy: 0.9371
o  Epoch 5/16
o  accuracy: 0.9458
```

- Number of parameters:

```
o Total params: 312,674
o Trainable params: 312,674
```

- Avg_time in each epoch for training data = 55s
- Testing time = 12s

15.Model 15:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="tanh"),
    layers.Dense(128,activation="tanh"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 100.00%
- Final accuracy on test = 97.69%
- Accuracy in first 5 epochs

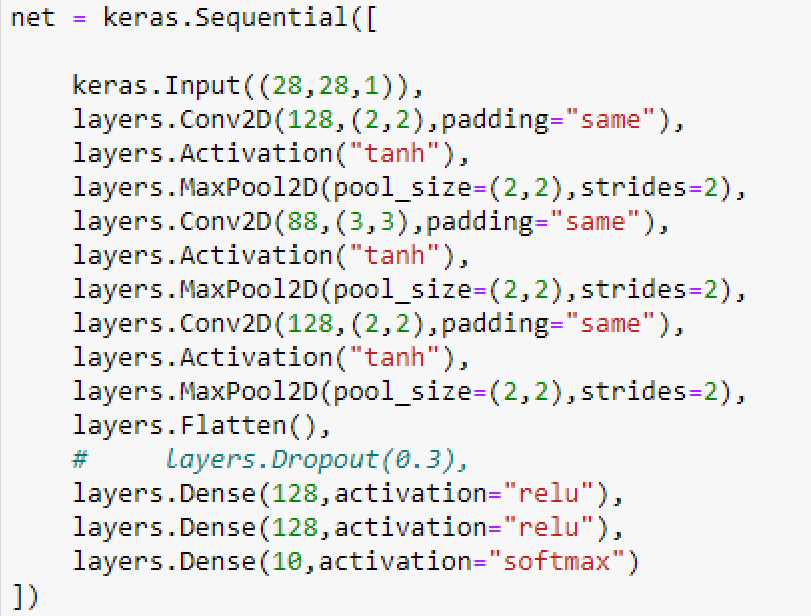
- `Epoch 1/16`
- `accuracy: 0.7509`
- `Epoch 2/16`
- `accuracy: 0.9215`
- `Epoch 3/16`
- `accuracy: 0.9460`
- `Epoch 4/16`
- `accuracy: 0.9536`
- `Epoch 5/16`
- `accuracy: 0.9637`

- Number of parameters:

```
o Total params: 312,674
o Trainable params: 312,674
```

- Avg_time in each epoch for training data = 48s
- Testing time = 10s

16.Model 16:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("tanh"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="relu"),
    layers.Dense(128,activation="relu"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 97.14%
- Final accuracy on test = 97.14%
- Accuracy in first 5 epochs

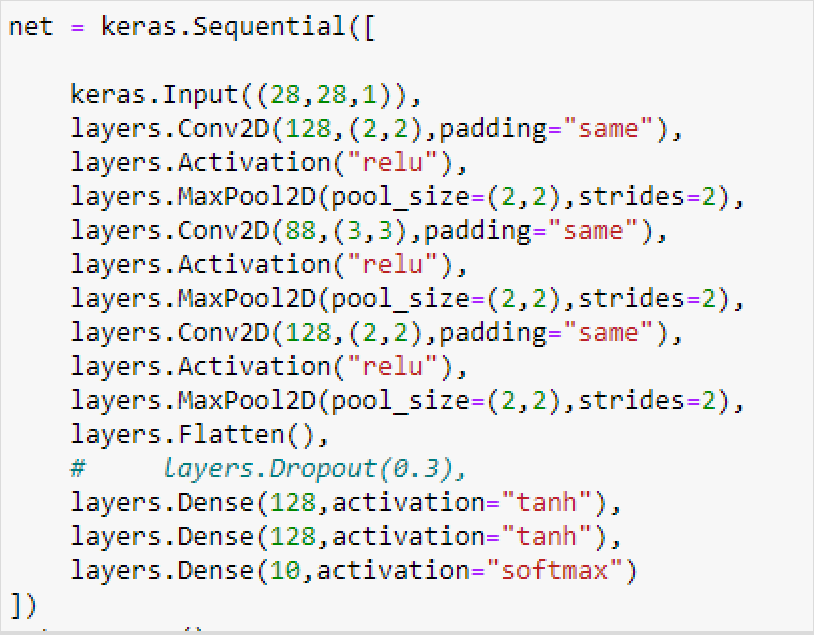
```
• Epoch 1/16
• accuracy: 0.5183
• Epoch 2/16
• accuracy: 0.8005
• Epoch 3/16
• accuracy: 0.8793
• Epoch 4/16
• accuracy: 0.9140
• Epoch 5/16
• accuracy: 0.9331
```

- Number of parameters:

```
o Total params: 312,674
o Trainable params: 312,674
```

- Avg_time in each epoch for training data = 51s
- Testing time = 11s

17.Model 17:



```
net = keras.Sequential([

    keras.Input((28,28,1)),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(88,(3,3),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Conv2D(128,(2,2),padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D(pool_size=(2,2),strides=2),
    layers.Flatten(),
    #     layers.Dropout(0.3),
    layers.Dense(128,activation="tanh"),
    layers.Dense(128,activation="tanh"),
    layers.Dense(10,activation="softmax")
])
```

- Final accuracy on train = 100.00%
- Final accuracy on test = 97.69%
- Accuracy in first 5 epochs
- Epoch 1/16
- accuracy: 0.7509
- Epoch 2/16
- accuracy: 0.9215
- Epoch 3/16
- accuracy: 0.9460
- Epoch 4/16
- accuracy: 0.9536
- Epoch 5/16
- accuracy: 0.9637

- Number of parameters:
    - Total params: 312,674
    - Trainable params: 312,674
- Avg_time in each epoch for training data = 48s
- Testing time = 10s