Project Presentation

Architecture used in the Paper::

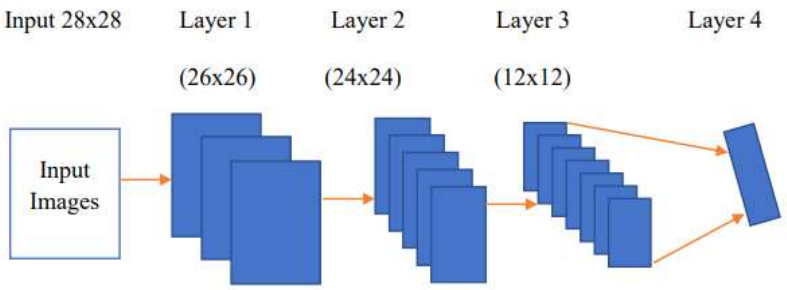


Figure 1: Architecture of Convolutional Neural Network (CNN)

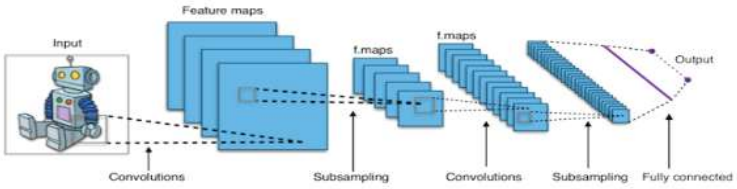


Figure. 2 Typical CNN Architecture

Datasets for it : **MNIST dataset**

dataset link :
https://www.tensorflow.org/datasets/catalog/mnist
It includes 70000 image for (Training & Testing)

number of classes : 10

dimension of images = 28*28

number of traning : 60000

number of testing : 10000

Our Model :

[69]:
```
model = Sequential()


model.add(Conv2D (32 ,(3,3) ,kernel_regularizer=regularizers.l2(0.0001), input_shape = x_train.shape [1: ]))
model.add(Activation ( "relu" ))
model.add(MaxPooling2D (pool_size = (2,2)))

model.add(Conv2D (64 ,(3,3),kernel_regularizer=regularizers.l2(0.0001)))
model.add(Activation ( "relu" ))
model.add(MaxPooling2D (pool_size = (2,2)))

model.add(Conv2D (64 ,(3,3),kernel_regularizer=regularizers.l2(0.0001)))
model.add(Activation ( "relu" ))
model.add(MaxPooling2D (pool_size = (2,2)))

model.add (Flatten())
model.add (Dense (64))
model.add (Activation ("relu"))

model.add (Dense (32))
model.add (Activation ("relu"))

model.add (Dense (10))
model.add (Activation ("softmax"))
```

model.summary:

In [70]:
```
model.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_14 (Conv2D)           (None, 26, 26, 32)        320
_________________________________________________________________
activation_21 (Activation)   (None, 26, 26, 32)        0
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 11, 11, 64)        18496
_________________________________________________________________
activation_22 (Activation)   (None, 11, 11, 64)        0
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 3, 3, 64)          36928
_________________________________________________________________
activation_23 (Activation)   (None, 3, 3, 64)          0
_________________________________________________________________
max_pooling2d_13 (MaxPooling (None, 1, 1, 64)          0
_________________________________________________________________
flatten_3 (Flatten)          (None, 64)                0
_________________________________________________________________
dense_9 (Dense)              (None, 64)                4160
_________________________________________________________________
activation_24 (Activation)   (None, 64)                0
_________________________________________________________________
dense_10 (Dense)             (None, 32)                2080
_________________________________________________________________
activation_25 (Activation)   (None, 32)                0
_________________________________________________________________
dense_11 (Dense)             (None, 10)                330
_________________________________________________________________
activation_26 (Activation)   (None, 10)                0
```

Hyperparameters:

optimizer = 'adam'

loss = ''sparse_categorical_crossentropy''

epochs = 5

batch_size = 128

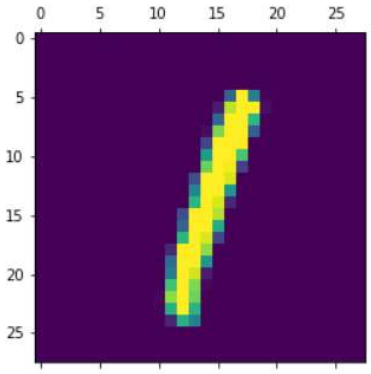
learning rate = 0.001

regularization = 0.0001

Result:::

this image is number 1 and he is predicted as num 1

```
In [115]: plt.matshow(x_test[5])
Out[115]: <matplotlib.image.AxesImage at 0x1bdb751c940>
```




```
In [116]: y_predicted = model.predict(x_test)

In [117]: np.argmax(y_predicted[5])
Out[117]: 1

In [118]: y_test[5]
Out[118]: 1
```

Accuracy::: 98%

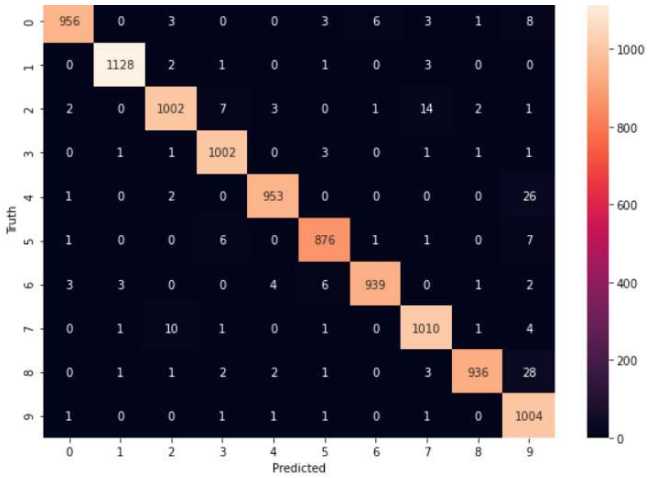
```
In [72]: hist = model.fit (x_train , y_train , batch_size = 128 , epochs = 5)

Epoch 1/5
469/469 [==============================] - 13s 28ms/step - loss: 0.4025 - accuracy: 0.8778
Epoch 2/5
469/469 [==============================] - 13s 29ms/step - loss: 0.1301 - accuracy: 0.9648
Epoch 3/5
469/469 [==============================] - 13s 28ms/step - loss: 0.1005 - accuracy: 0.9745
Epoch 4/5
469/469 [==============================] - 13s 28ms/step - loss: 0.0847 - accuracy: 0.9791
Epoch 5/5
469/469 [==============================] - 13s 29ms/step - loss: 0.0755 - accuracy: 0.9825

In [73]: model.evaluate(x_test , y_test )

313/313 [==============================] - 1s 3ms/step - loss: 0.0810 - accuracy: 0.9806
Out[73]: [0.08100221306085587, 0.9805999994277954]
```

Confusion matrix:



loss Curve:

```
In [82]: plt.figure(figsize = (10,7))
         plt.plot(xc , train_loss)
         plt.xlabel('num of epochs')
         plt.ylabel('loss')
```

```
Out[82]: Text(0, 0.5, 'loss')
```