NAME: AYUSH CHANCHAL

SAP ID: 500097569

BATCH: B6

EXPERIMENT: 06

Q. What is CNN?

: Convolutional Neural Network is a Deep Learning algorithm specially designed for working with Images and videos. It takes images as inputs, extracts and learns the features of the image, and classifies them based on the learned features. CNNs are particularly useful for finding patterns in images to recognize objects, classes, and categories. They can also be quite effective for classifying audio, time-series, and signal data.

Q. Components of CNN

: There are four major components of CNN which are as follows-

* **Input layer-** As the name says, it’s our input image and can be Grayscale or RGB. Every image is made up of pixels that range from 0 to 255. We need to normalize them i.e., convert the range between 0 to 1 before passing it to the model. Below is the example of an input image of size 4\*4 and has 3 channels i.e., RGB and pixel values.
* **Convolution layer + Activation function-**The convolution layer is the layer where the filter is applied to our input image to extract or A filter is applied to the image multiple times and creates a feature map which helps in classifying the input image.

The activation function layer helps the network learn non-linear relationships between the input and output. It is responsible for introducing non-linearity into the network and allowing it to model complex patterns and relationships in the data.

* **Pooling layer-** The pooling layer is applied after the Convolutional layer and is used to reduce the dimensions of

the feature map which helps in preserving the important information or features of the input image and reduces the computation time.

* **Fully Connected Layer-** Fully connected layer is where final classification takes place, but for this, we need the input to

be in one dimension hence, first performing flattening. Then few fully connected layers are used with the last layer having exactly the same no. of neurons as the total no. of classes we have for classification.

Source code:

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D

from tensorflow.keras.layers import MaxPool2D

from tensorflow.keras.layers import Flatten

from tensorflow.keras.layers import Dropout

from tensorflow.keras.layers import Dense

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import accuracy\_score

(x\_train,y\_train),(x\_test,y\_test)=mnist.load\_data()

x\_train=x\_train.reshape((x\_train.shape[0],x\_train.shape[1],x\_train.shape[2],1))

x\_test=x\_test.reshape((x\_test.shape[0],x\_test.shape[1],x\_test.shape[2],1))

print(x\_train.shape)

print(x\_test.shape)

model=Sequential()

model.add(Conv2D(32,(3,3),activation="relu",input\_shape=(28,28,1)))

model.add(MaxPool2D(2,2))

model.add(Flatten())

model.add(Dense(100,activation="relu"))

model.add(Dense(10,activation="softmax"))

model.compile(loss="sparse\_categorical\_crossentropy",optimizer="adam",metrics=["accuracy"])

model.fit(x\_train,y\_train,epochs=10)

y\_pred=model.predict(x\_test)

a=y\_pred.argmax(axis=1)

cm=confusion\_matrix(a,y\_test)

print(cm)

acc=accuracy\_score(a,y\_test)

print(f"Accuracy={round(acc\*100,2)}%")

Screenshot:

