**Program No: 13**

**Date:02/02/2022**

**Aim:**Programs on convolutional neural network to classify images from any standard dataset in the public domain.

Program

import numpy as np import pandas as pd

import matplotlib.pyplot as plt import tensorflow as tf

from tensorflow import keras np.random.seed(42) fashion\_mnist=keras.datasets.fashion\_mnist

(x\_train,y\_train),(x\_test,y\_test)=fashion\_mnist.load\_data() print(x\_train.shape,x\_test.shape)

x\_train=x\_train/255.0 x\_test=x\_test/255.0 plt.imshow(x\_train[1],cmap='binary') plt.show()

np.unique(y\_test)

class\_names=['T-shirt/Top','Trouser','Pullover','Dress','Coat','Sandal','Shirt','Sneaker','Bag','Ankle Boot']

n\_rows=5 n\_cols=10

plt.figure(figsize=(n\_cols \* 1.4,n\_rows \* 1.6)) for row in range(n\_rows):

for col in range(n\_cols): index=n\_cols \* row +col plt.subplot(n\_rows,n\_cols,index+1)

plt.imshow(x\_train[index],cmap='binary',interpolation='nearest') plt.axis('off')

plt.title(class\_names[y\_train[index]]) plt.show() model\_CNN=keras.models.Sequential()

model\_CNN.add(keras.layers.Conv2D(filters=32,kernel\_size=7,padding='same',activation='relu', input\_shape=[28,28,1]))

model\_CNN.add(keras.layers.MaxPooling2D(pool\_size=2)) model\_CNN.add(keras.layers.Conv2D(filters=64,kernel\_size=3,padding='same',activation='relu'

))

model\_CNN.add(keras.layers.MaxPooling2D(pool\_size=2))

model\_CNN.add(keras.layers.Conv2D(filters=32,kernel\_size=3,padding='same',activation='relu'

))

model\_CNN.add(keras.layers.MaxPooling2D(pool\_size=2)) model\_CNN.summary() model\_CNN.add(keras.layers.Flatten())

model\_CNN.add(keras.layers.Dense(units=128,activation='relu')) model\_CNN.add(keras.layers.Dense(units=64,activation='relu')) model\_CNN.add(keras.layers.Dense(units=10,activation='softmax')) model\_CNN.summary()

model\_CNN.compile(loss='sparse\_categorical\_crossentropy',optimizer='adam',metrics=['accurac y'])

x\_train=x\_train[...,np.newaxis] x\_test=x\_test[...,np.newaxis]

history\_CNN=model\_CNN.fit(x\_train,y\_train,epochs=2,validation\_split=0.1) pd.DataFrame(history\_CNN.history).plot()

plt.grid(True) plt.xlabel('epochs') plt.ylabel('loss/accuracy')

plt.title('Training and validation plot') plt.show()

test\_loss,test\_accuracy=model\_CNN.evaluate(x\_test,y\_test) print('Test Loss:{}','Test Accuracy:{}'.format(test\_loss,test\_accuracy))

OUTPUT











