# Practical no.9

**Aim:** Classification of images of clothing using Tensorflow (Fashion MNIST dataset)

## Theory

Image classification is a common task in machine learning where the goal is to classify an image into one of several pre-defined categories. One popular dataset for image classification is the Fashion-MNIST dataset, which contains 70,000 gray scale images of 10 different types of clothing items such as t-shirts, dresses, and shoes.

## Material

* tensflow

## Program

import tensorflow as tf

from tensorflow.keras.datasets import fashion\_mnist # Load the Fashion MNIST dataset

(train\_images, train\_labels), (test\_images, test\_labels) = fashion\_mnist.load\_data() # Define the class names

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

# Preprocess the data

train\_images = train\_images / 255.0 test\_images = test\_images / 255.0

# Define the model architecture model = tf.keras.Sequential([

tf.keras.layers.Flatten(input\_shape=(28, 28)), tf.keras.layers.Dense(128, activation='relu'), tf.keras.layers.Dense(10)

])

# Compile the model model.compile(optimizer='adam',

loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True), metrics=['accuracy'])

# Train the model

history = model.fit(train\_images, train\_labels, epochs=10,validation\_data=(test\_images, test\_labels))

# Evaluate the model on the test set

test\_loss, test\_acc = model.evaluate(test\_images, test\_labels, verbose=2) print('\nTest accuracy:', test\_acc)

# Make predictions on new data predictions = model.predict(test\_images) # Print a few predictions

import numpy as np for i in range(5):

predicted\_label = np.argmax(predictions[i]) true\_label = test\_labels[i]

print("Predicted: {}, Actual:

{}".format(class\_names[predicted\_label],class\_names[true\_label]))

# Output:

