import tensorflow as tf #import tensorflow

mnist = tf.keras.datasets.mnist # get MNIST datasets

(x\_train, y\_train),(x\_test, y\_test) = mnist.load\_data() # load and spit into train-test data

x\_train, x\_test = x\_train / 255.0, x\_test / 255.0 # normalize the data

model = tf.keras.models.Sequential([ # Create a Linear stack of layers, list of layers to add to the model

tf.keras.layers.Flatten(), # Flattens the input. To shape the input layer.

tf.keras.layers.Dense(512, activation=tf.nn.relu), # Dense implements the operation: output = activation(dot(input, kernel) + bias), like broadcast on each element. Use ReLu activation function.

tf.keras.layers.Dropout(0.2), # Dropout consists in randomly setting a fraction rate of input units to 0 at each update during training time, which helps prevent over-fitting.

tf.keras.layers.Dense(10, activation=tf.nn.softmax) # Make dense connection between hidden and output and apply softmax activation function.

])

model.compile(optimizer='adam',loss='sparse\_categorical\_crossentropy', metrics=['accuracy']) # compile the model created

model.fit(x\_train, y\_train, epochs=5) # Trains the model for a fixed number of epochs (iterations on a dataset).

model.evaluate(x\_test, y\_test) #Returns the loss value & metrics values for the model in test mode.