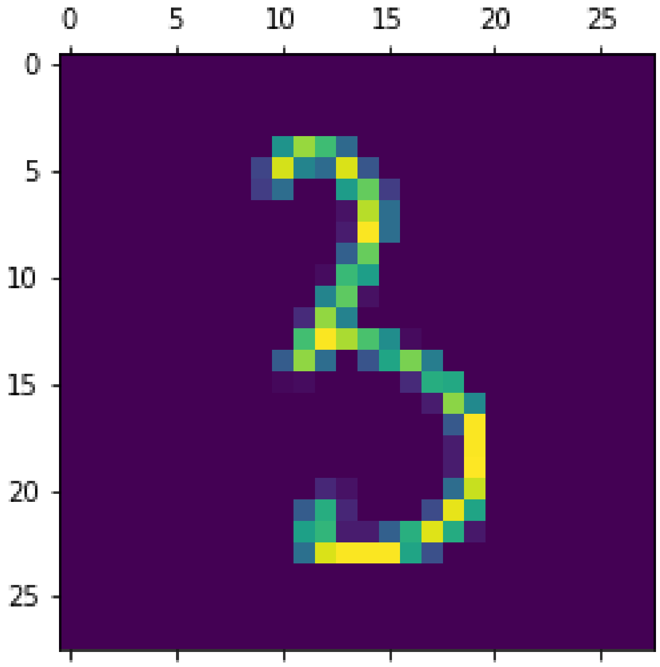
```
"""
DL Lab Assignment-2
Problem Statement:
   Implementing Feedforward neural networks with Keras and TensorFlow
   a. Import the necessary packages
   b. Load the training and testing data (MNIST/CIFAR10)
   c. Define the network architecture using Keras
   d. Train the model using SGD
   e. Evaluate the network
   f. Plot the training loss and accuracy
"""

# a. IMPORTING NECESSARY PACKAGES ->
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import random

# b. LOAD THE TRAINING AND TESTING DATA (MNIST)  ->
mnist = tf.keras.datasets.mnist          # Importing MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()    # Splitting it into training and testing data
plt.matshow(x_train[1])
plt.imshow(-x_train[0], cmap="gray")

x_train = x_train / 255
x_test = x_test / 255

# c. DEFINE THE NETWORK ARCHITECTURE USING KERAS ->
model = keras.Sequential([
keras.layers.Flatten(input_shape=(28, 28)),
keras.layers.Dense(128, activation="relu"),
keras.layers.Dense(10, activation="softmax")
])

model.summary()

# d. TRAIN THE MODEL USING SGD ->
model.compile(optimizer="sgd",
loss="sparse_categorical_crossentropy",
metrics=['accuracy'])

history=model.fit(x_train,
y_train,validation_data=(x_test,y_test),epochs=10)



# e. EVALUATE THE NETWORK
test_loss,test_acc=model.evaluate(x_test,y_test)
print("Loss=%.3f" %test_loss)
print("Accuracy=%.3f" %test_acc)

n=random.randint(0,9999)
plt.imshow(x_test[n])
plt.show()
x_train
x_test

predicted_value=model.predict(x_test)
plt.imshow(x_test[n])
plt.show()

print(predicted_value[n])
```

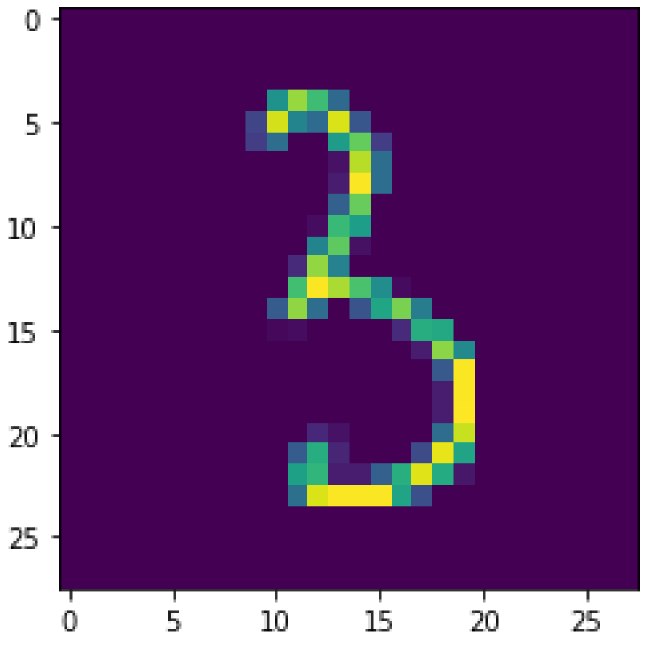
```
# f. PLOT THE TRAINING LOSS AND ACCURACY ->


   #  Plotting The Training Accuracy :-
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()



   # Plotting The Training Loss :-
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

OUTPUTS

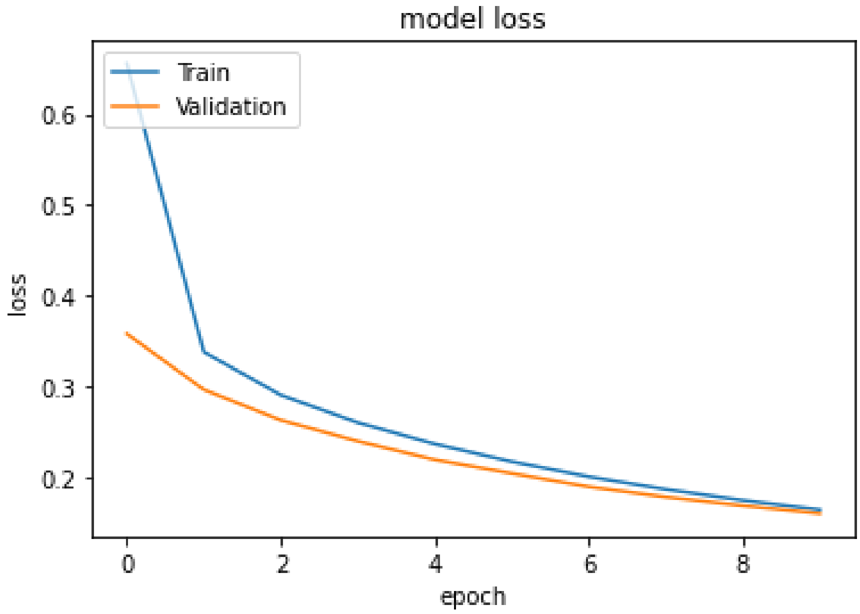
- Training



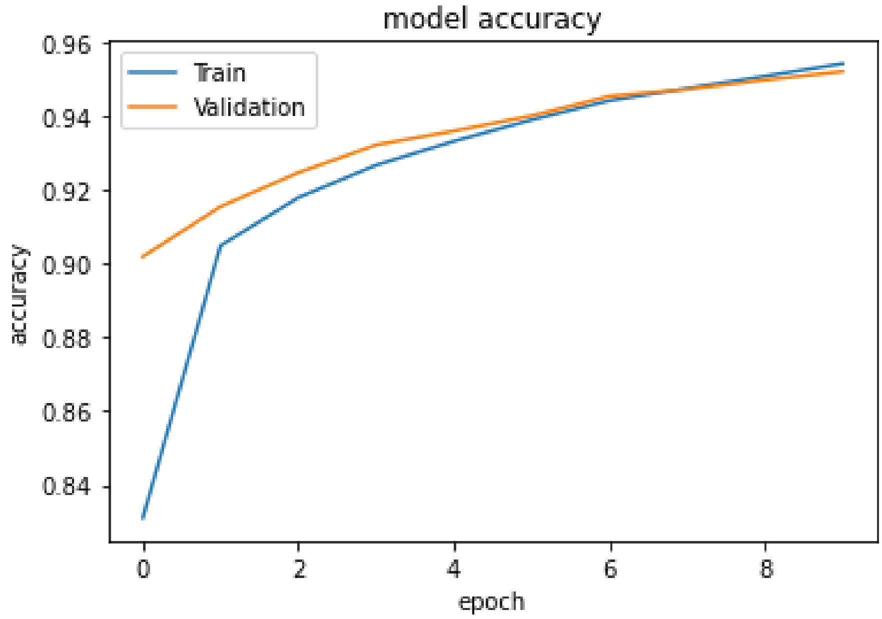
- Testing



- Model Loss



- Model Accuracy


model accuracy
Train
Validation
accuracy
epoch