

In [2]: #Assignment No: 2  
#Title: Implementing Feedforward neural networks with Keras and TensorFlow

In [5]: from sklearn.preprocessing import LabelBinarizer  
from sklearn.metrics import classification\_report  
from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense  
from tensorflow.keras.optimizers import Adam  
from tensorflow.keras.datasets import mnist

In [6]: import matplotlib.pyplot as plt  
import numpy as np

In [7]: print("[INFO] accessing MNIST...")  
  
[INFO] accessing MNIST...

In [8]: ((trainX,trainY), (testX,testY)) = mnist.load\_data()  
  
In [9]: trainX.shape  
  
Out[9]: (60000, 28, 28)

In [10]: trainX = trainX.reshape((trainX.shape[0], 28\*28 \* 1))  
testX = testX.reshape((testX.shape[0], 28 \* 28 \* 1))

In [11]: trainX = trainX.astype("float32") / 255.0  
testX= testX.astype("float32") / 255.0

In [12]: lb=LabelBinarizer()  
trainY= lb.fit\_transform(trainY)  
testY = lb.transform(testY)

In [13]: model = Sequential()

In [14]: model.add(Dense (256, input\_shape=(784,), activation="relu"))  
model.add(Dense (128, activation="relu"))  
model.add(Dense (64, activation="relu"))  
model.add(Dense (10, activation="softmax"))  
  
C:\Users\yadav\anaconda3\Lib\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass an `input\_shape`/`input\_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.  
super().\_\_init\_\_(activity\_regularizer=activity\_regularizer, \*\*kwargs)

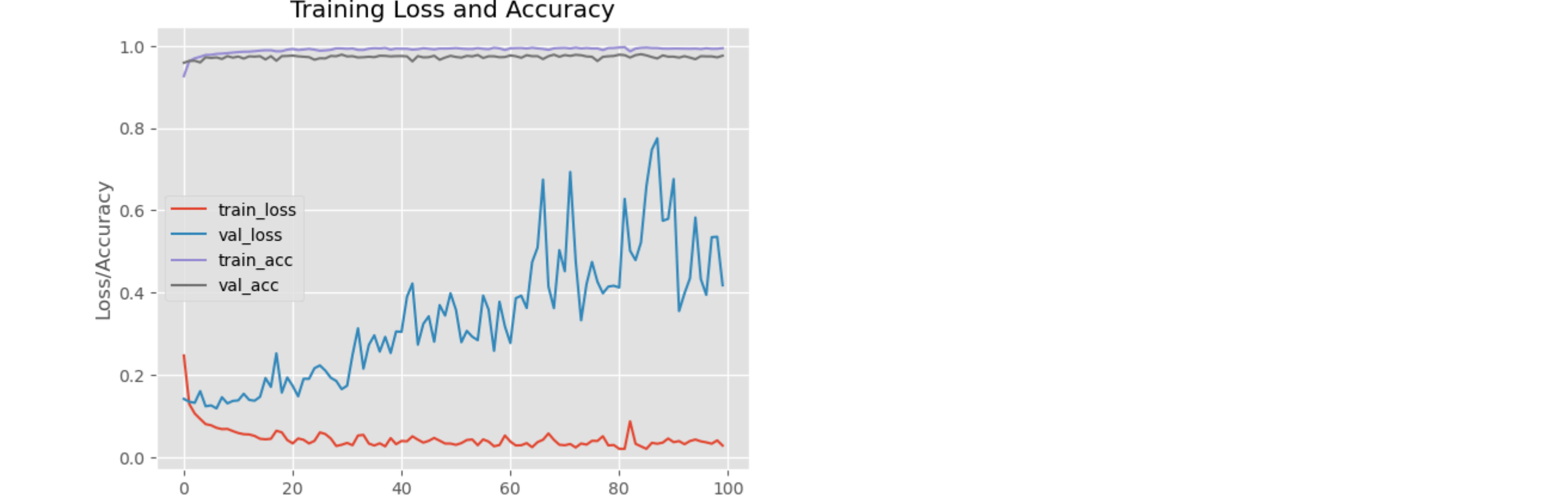
In [15]: print("[INFO] training network...")  
Adm = Adam(0.01)  
model.compile(loss="categorical\_crossentropy", optimizer = Adm, metrics=["accuracy"])  
H = model.fit(trainX, trainY, validation\_data=(testX, testY), epochs=100, batch\_size=128)  
  
[INFO] training network...  
Epoch 1/100  
469/469 — 19s 17ms/step - accuracy: 0.8688 - loss: 0.4275 - val\_accuracy: 0.9588 - val\_loss: 0.1417  
Epoch 2/100  
469/469 — 7s 15ms/step - accuracy: 0.9613 - loss: 0.1337 - val\_accuracy: 0.9632 - val\_loss: 0.1348  
Epoch 3/100  
469/469 — 11s 16ms/step - accuracy: 0.9706 - loss: 0.1013 - val\_accuracy: 0.9638 - val\_loss: 0.1322  
Epoch 4/100  
469/469 — 11s 17ms/step - accuracy: 0.9762 - loss: 0.0823 - val\_accuracy: 0.9596 - val\_loss: 0.1607  
Epoch 5/100  
469/469 — 11s 17ms/step - accuracy: 0.9766 - loss: 0.0826 - val\_accuracy: 0.9722 - val\_loss: 0.1238  
Epoch 6/100  
469/469 — 10s 16ms/step - accuracy: 0.9790 - loss: 0.0737 - val\_accuracy: 0.9705 - val\_loss: 0.1260  
Epoch 7/100  
469/469 — 10s 15ms/step - accuracy: 0.9811 - loss: 0.0664 - val\_accuracy: 0.9719 - val\_loss: 0.1185  
Epoch 8/100  
469/469 — 7s 15ms/step - accuracy: 0.9824 - loss: 0.0624 - val\_accuracy: 0.9681 - val\_loss: 0.1458  
Epoch 9/100  
469/469 — 8s 16ms/step - accuracy: 0.9815 - loss: 0.0695 - val\_accuracy: 0.9750 - val\_loss: 0.1308  
Epoch 10/100  
469/469 — 8s 16ms/step - accuracy: 0.9841 - loss: 0.0599 - val\_accuracy: 0.9713 - val\_loss: 0.1368  
Epoch 11/100  
469/469 — 8s 15ms/step - accuracy: 0.9842 - loss: 0.0613 - val\_accuracy: 0.9740 - val\_loss: 0.1381  
Epoch 12/100  
469/469 — 11s 15ms/step - accuracy: 0.9869 - loss: 0.0502 - val\_accuracy: 0.9691 - val\_loss: 0.1543  
Epoch 13/100  
469/469 — 11s 17ms/step - accuracy: 0.9864 - loss: 0.0514 - val\_accuracy: 0.9745 - val\_loss: 0.1393  
Epoch 14/100  
469/469 — 10s 15ms/step - accuracy: 0.9875 - loss: 0.0511 - val\_accuracy: 0.9738 - val\_loss: 0.1374  
Epoch 15/100  
469/469 — 8s 17ms/step - accuracy: 0.9889 - loss: 0.0403 - val\_accuracy: 0.9748 - val\_loss: 0.1469  
Epoch 16/100  
469/469 — 9s 19ms/step - accuracy: 0.9903 - loss: 0.0416 - val\_accuracy: 0.9669 - val\_loss: 0.1924  
Epoch 17/100  
469/469 — 9s 16ms/step - accuracy: 0.9900 - loss: 0.0393 - val\_accuracy: 0.9749 - val\_loss: 0.1709  
Epoch 18/100  
469/469 — 11s 16ms/step - accuracy: 0.9882 - loss: 0.0516 - val\_accuracy: 0.9638 - val\_loss: 0.2525  
Epoch 19/100  
469/469 — 10s 14ms/step - accuracy: 0.9860 - loss: 0.0696 - val\_accuracy: 0.9749 - val\_loss: 0.1570  
Epoch 20/100  
469/469 — 11s 14ms/step - accuracy: 0.9906 - loss: 0.0438 - val\_accuracy: 0.9754 - val\_loss: 0.1937  
Epoch 21/100  
469/469 — 7s 14ms/step - accuracy: 0.9929 - loss: 0.0323 - val\_accuracy: 0.9763 - val\_loss: 0.1733  
Epoch 22/100  
469/469 — 11s 15ms/step - accuracy: 0.9905 - loss: 0.0415 - val\_accuracy: 0.9745 - val\_loss: 0.1481  
Epoch 23/100  
469/469 — 11s 17ms/step - accuracy: 0.9917 - loss: 0.0359 - val\_accuracy: 0.9735 - val\_loss: 0.1906  
Epoch 24/100  
469/469 — 10s 16ms/step - accuracy: 0.9928 - loss: 0.0355 - val\_accuracy: 0.9726 - val\_loss: 0.1905  
Epoch 25/100  
469/469 — 7s 15ms/step - accuracy: 0.9921 - loss: 0.0346 - val\_accuracy: 0.9663 - val\_loss: 0.2163  
Epoch 26/100  
469/469 — 8s 16ms/step - accuracy: 0.9874 - loss: 0.0641 - val\_accuracy: 0.9698 - val\_loss: 0.2231  
Epoch 27/100  
469/469 — 8s 17ms/step - accuracy: 0.9901 - loss: 0.0464 - val\_accuracy: 0.9694 - val\_loss: 0.2105  
Epoch 28/100  
469/469 — 9s 18ms/step - accuracy: 0.9902 - loss: 0.0509 - val\_accuracy: 0.9754 - val\_loss: 0.1931  
Epoch 29/100  
469/469 — 10s 16ms/step - accuracy: 0.9945 - loss: 0.0245 - val\_accuracy: 0.9747 - val\_loss: 0.1854  
Epoch 30/100  
469/469 — 10s 20ms/step - accuracy: 0.9942 - loss: 0.0274 - val\_accuracy: 0.9787 - val\_loss: 0.1654  
Epoch 31/100  
469/469 — 9s 16ms/step - accuracy: 0.9944 - loss: 0.0257 - val\_accuracy: 0.9743 - val\_loss: 0.1738  
Epoch 32/100  
469/469 — 8s 16ms/step - accuracy: 0.9936 - loss: 0.0311 - val\_accuracy: 0.9748 - val\_loss: 0.2492  
Epoch 33/100  
469/469 — 9s 17ms/step - accuracy: 0.9925 - loss: 0.0435 - val\_accuracy: 0.9717 - val\_loss: 0.3133  
Epoch 34/100  
469/469 — 11s 17ms/step - accuracy: 0.9909 - loss: 0.0510 - val\_accuracy: 0.9723 - val\_loss: 0.2153  
Epoch 35/100  
469/469 — 11s 16ms/step - accuracy: 0.9932 - loss: 0.0321 - val\_accuracy: 0.9735 - val\_loss: 0.2732  
Epoch 36/100  
469/469 — 10s 16ms/step - accuracy: 0.9954 - loss: 0.0218 - val\_accuracy: 0.9725 - val\_loss: 0.2963  
Epoch 37/100  
469/469 — 11s 16ms/step - accuracy: 0.9930 - loss: 0.0379 - val\_accuracy: 0.9759 - val\_loss: 0.2567  
Epoch 38/100  
469/469 — 11s 16ms/step - accuracy: 0.9949 - loss: 0.0234 - val\_accuracy: 0.9756 - val\_loss: 0.2925  
Epoch 39/100  
469/469 — 11s 16ms/step - accuracy: 0.9923 - loss: 0.0383 - val\_accuracy: 0.9743 - val\_loss: 0.2534  
Epoch 40/100  
469/469 — 7s 14ms/step - accuracy: 0.9932 - loss: 0.0322 - val\_accuracy: 0.9750 - val\_loss: 0.3056  
Epoch 41/100  
469/469 — 12s 17ms/step - accuracy: 0.9936 - loss: 0.0377 - val\_accuracy: 0.9752 - val\_loss: 0.3049  
Epoch 42/100  
469/469 — 8s 16ms/step - accuracy: 0.9952 - loss: 0.0259 - val\_accuracy: 0.9744 - val\_loss: 0.3898  
Epoch 43/100  
469/469 — 11s 17ms/step - accuracy: 0.9899 - loss: 0.0524 - val\_accuracy: 0.9623 - val\_loss: 0.4223  
Epoch 44/100  
469/469 — 12s 18ms/step - accuracy: 0.9901 - loss: 0.0498 - val\_accuracy: 0.9751 - val\_loss: 0.2735  
Epoch 45/100  
469/469 — 8s 17ms/step - accuracy: 0.9948 - loss: 0.0237 - val\_accuracy: 0.9719 - val\_loss: 0.3241  
Epoch 46/100  
469/469 — 8s 17ms/step - accuracy: 0.9918 - loss: 0.0528 - val\_accuracy: 0.9724 - val\_loss: 0.3426  
Epoch 47/100  
469/469 — 12s 18ms/step - accuracy: 0.9919 - loss: 0.0442 - val\_accuracy: 0.9759 - val\_loss: 0.2807  
Epoch 48/100  
469/469 — 12s 21ms/step - accuracy: 0.9949 - loss: 0.0305 - val\_accuracy: 0.9662 - val\_loss: 0.3699  
Epoch 49/100  
469/469 — 8s 17ms/step - accuracy: 0.9928 - loss: 0.0319 - val\_accuracy: 0.9715 - val\_loss: 0.3443  
Epoch 50/100  
469/469 — 11s 18ms/step - accuracy: 0.9935 - loss: 0.0373 - val\_accuracy: 0.9757 - val\_loss: 0.3984  
Epoch 51/100  
469/469 — 8s 17ms/step - accuracy: 0.9948 - loss: 0.0271 - val\_accuracy: 0.9730 - val\_loss: 0.3583  
Epoch 52/100  
469/469 — 11s 18ms/step - accuracy: 0.9943 - loss: 0.0278 - val\_accuracy: 0.9714 - val\_loss: 0.2795  
Epoch 53/100  
469/469 — 10s 15ms/step - accuracy: 0.9921 - loss: 0.0399 - val\_accuracy: 0.9756 - val\_loss: 0.3071  
Epoch 54/100  
469/469 — 8s 17ms/step - accuracy: 0.9913 - loss: 0.0507 - val\_accuracy: 0.9743 - val\_loss: 0.2929  
Epoch 55/100  
469/469 — 11s 17ms/step - accuracy: 0.9937 - loss: 0.0338 - val\_accuracy: 0.9777 - val\_loss: 0.2845  
Epoch 56/100  
469/469 — 13s 22ms/step - accuracy: 0.9959 - loss: 0.0235 - val\_accuracy: 0.9705 - val\_loss: 0.3928  
Epoch 57/100  
469/469 — 9s 19ms/step - accuracy: 0.9913 - loss: 0.0432 - val\_accuracy: 0.9746 - val\_loss: 0.3588  
Epoch 58/100  
469/469 — 13s 23ms/step - accuracy: 0.9948 - loss: 0.0338 - val\_accuracy: 0.9746 - val\_loss: 0.2586  
Epoch 59/100  
469/469 — 20s 21ms/step - accuracy: 0.9937 - loss: 0.0277 - val\_accuracy: 0.9723 - val\_loss: 0.3780  
Epoch 60/100  
469/469 — 10s 19ms/step - accuracy: 0.9898 - loss: 0.0591 - val\_accuracy: 0.9728 - val\_loss: 0.3179  
Epoch 61/100  
469/469 — 13s 23ms/step - accuracy: 0.9935 - loss: 0.0399 - val\_accuracy: 0.9767 - val\_loss: 0.2775  
Epoch 62/100  
469/469 — 20s 22ms/step - accuracy: 0.9944 - loss: 0.0294 - val\_accuracy: 0.9747 - val\_loss: 0.3865  
Epoch 63/100  
469/469 — 12s 23ms/step - accuracy: 0.9952 - loss: 0.0234 - val\_accuracy: 0.9710 - val\_loss: 0.3926  
Epoch 64/100  
469/469 — 11s 22ms/step - accuracy: 0.9922 - loss: 0.0407 - val\_accuracy: 0.9772 - val\_loss: 0.3626  
Epoch 65/100  
469/469 — 11s 22ms/step - accuracy: 0.9952 - loss: 0.0241 - val\_accuracy: 0.9748 - val\_loss: 0.4742  
Epoch 66/100  
469/469 — 9s 18ms/step - accuracy: 0.9950 - loss: 0.0296 - val\_accuracy: 0.9749 - val\_loss: 0.5098  
Epoch 67/100  
469/469 — 12s 20ms/step - accuracy: 0.9934 - loss: 0.0309 - val\_accuracy: 0.9679 - val\_loss: 0.6749  
Epoch 68/100  
469/469 — 10s 16ms/step - accuracy: 0.9899 - loss: 0.0606 - val\_accuracy: 0.9748 - val\_loss: 0.4150  
Epoch 69/100  
469/469 — 8s 16ms/step - accuracy: 0.9935 - loss: 0.0406 - val\_accuracy: 0.9785 - val\_loss: 0.3622  
Epoch 70/100  
469/469 — 9s 18ms/step - accuracy: 0.9945 - loss: 0.0325 - val\_accuracy: 0.9734 - val\_loss: 0.5036  
Epoch 71/100  
469/469 — 11s 18ms/step - accuracy: 0.9947 - loss: 0.0329 - val\_accuracy: 0.9775 - val\_loss: 0.4521  
Epoch 72/100  
469/469 — 11s 17ms/step - accuracy: 0.9935 - loss: 0.0355 - val\_accuracy: 0.9755 - val\_loss: 0.6935  
Epoch 73/100  
469/469 — 11s 16ms/step - accuracy: 0.9947 - loss: 0.0292 - val\_accuracy: 0.9782 - val\_loss: 0.4777  
Epoch 74/100  
469/469 — 8s 15ms/step - accuracy: 0.9945 - loss: 0.0292 - val\_accuracy: 0.9771 - val\_loss: 0.3328  
Epoch 75/100  
469/469 — 11s 17ms/step - accuracy: 0.9950 - loss: 0.0298 - val\_accuracy: 0.9742 - val\_loss: 0.4212  
Epoch 76/100  
469/469 — 13s 21ms/step - accuracy: 0.9943 - loss: 0.0354 - val\_accuracy: 0.9734 - val\_loss: 0.4747  
Epoch 77/100  
469/469 — 8s 16ms/step - accuracy: 0.9940 - loss: 0.0372 - val\_accuracy: 0.9631 - val\_loss: 0.4266  
Epoch 78/100  
469/469 — 8s 16ms/step - accuracy: 0.9869 - loss: 0.0641 - val\_accuracy: 0.9733 - val\_loss: 0.3984  
Epoch 79/100  
469/469 — 8s 16ms/step - accuracy: 0.9935 - loss: 0.0316 - val\_accuracy: 0.9746 - val\_loss: 0.4148  
Epoch 80/100  
469/469 — 11s 17ms/step - accuracy: 0.9942 - loss: 0.0334 - val\_accuracy: 0.9754 - val\_loss: 0.4168  
Epoch 81/100  
469/469 — 8s 17ms/step - accuracy: 0.9962 - loss: 0.0180 - val\_accuracy: 0.9785 - val\_loss: 0.4125  
Epoch 82/100  
469/469 — 8s 17ms/step - accuracy: 0.9977 - loss: 0.0113 - val\_accuracy: 0.9774 - val\_loss: 0.6280  
Epoch 83/100  
469/469 — 8s 16ms/step - accuracy: 0.9876 - loss: 0.1063 - val\_accuracy: 0.9715 - val\_loss: 0.5019  
Epoch 84/100  
469/469 — 9s 13ms/step - accuracy: 0.9924 - loss: 0.0348 - val\_accuracy: 0.9774 - val\_loss: 0.4792  
Epoch 85/100  
469/469 — 7s 14ms/step - accuracy: 0.9947 - loss: 0.0307 - val\_accuracy: 0.9797 - val\_loss: 0.5222  
Epoch 86/100  
469/469 — 11s 16ms/step - accuracy: 0.9959 - loss: 0.0194 - val\_accuracy: 0.9768 - val\_loss: 0.6587  
Epoch 87/100  
469/469 — 10s 14ms/step - accuracy: 0.9950 - loss: 0.0298 - val\_accuracy: 0.9728 - val\_loss: 0.7475  
Epoch 88/100  
469/469 — 11s 14ms/step - accuracy: 0.9954 - loss: 0.0203 - val\_accuracy: 0.9693 - val\_loss: 0.7752  
Epoch 89/100  
469/469 — 7s 15ms/step - accuracy: 0.9920 - loss: 0.0437 - val\_accuracy: 0.9767 - val\_loss: 0.5747  
Epoch 90/100  
469/469 — 6s 13ms/step - accuracy: 0.9949 - loss: 0.0340 - val\_accuracy: 0.9736 - val\_loss: 0.5795  
Epoch 91/100  
469/469 — 11s 13ms/step - accuracy: 0.9930 - loss: 0.0464 - val\_accuracy: 0.9737 - val\_loss: 0.6761  
Epoch 92/100  
469/469 — 6s 13ms/step - accuracy: 0.9941 - loss: 0.0330 - val\_accuracy: 0.9714 - val\_loss: 0.3552  
Epoch 93/100  
469/469 — 7s 14ms/step - accuracy: 0.9927 - loss: 0.0355 - val\_accuracy: 0.9749 - val\_loss: 0.3983  
Epoch 94/100  
469/469 — 7s 14ms/step - accuracy: 0.9938 - loss: 0.0349 - val\_accuracy: 0.9714 - val\_loss: 0.4348  
Epoch 95/100  
469/469 — 10s 13ms/step - accuracy: 0.9930 - loss: 0.0410 - val\_accuracy: 0.9675 - val\_loss: 0.5827  
Epoch 96/100  
469/469 — 11s 14ms/step - accuracy: 0.9916 - loss: 0.0408 - val\_accuracy: 0.9749 - val\_loss: 0.4330  
Epoch 97/100  
469/469 — 11s 14ms/step - accuracy: 0.9940 - loss: 0.0317 - val\_accuracy: 0.9743 - val\_loss: 0.3946  
Epoch 98/100  
469/469 — 10s 13ms/step - accuracy: 0.9929 - loss: 0.0331 - val\_accuracy: 0.9743 - val\_loss: 0.5349  
Epoch 99/100  
469/469 — 11s 14ms/step - accuracy: 0.9922 - loss: 0.0460 - val\_accuracy: 0.9720 - val\_loss: 0.5359  
Epoch 100/100  
469/469 — 7s 14ms/step - accuracy: 0.9932 - loss: 0.0357 - val\_accuracy: 0.9760 - val\_loss: 0.4180

In [22]: print("[INFO] evaluating network...")  
predictions = model.predict(testX, batch\_size=128)  
print(classification\_report(testY.argmax(axis=1), predictions.argmax(axis=1), target\_names=[str(i) for i in range(10)]))

[INFO] evaluating network...  
79/79 — 1s 8ms/step  
precision recall f1-score support  
0 0.99 0.99 0.99 980  
1 0.99 0.98 0.99 1135  
2 0.98 0.97 0.98 1032  
3 0.92 0.98 0.95 1010  
4 0.97 0.98 0.98 982  
5 0.99 0.96 0.97 892  
6 0.99 0.98 0.98 958  
7 0.99 0.97 0.98 1028  
8 0.96 0.97 0.97 974  
9 0.98 0.97 0.97 1009  
  
accuracy 0.98 10000  
macro avg 0.98 0.98 0.98 10000  
weighted avg 0.98 0.98 0.98 10000

In [23]: plt.style.use("ggplot")  
plt.figure()  
plt.plot(np.arange(0, 100), H.history["loss"], label="train\_loss")  
plt.plot(np.arange(0, 100), H.history["val\_loss"], label="val\_loss")  
plt.plot(np.arange(0, 100), H.history["accuracy"], label="train\_acc")  
plt.plot(np.arange(0, 100), H.history["val\_accuracy"], label="val\_acc")  
plt.title("Training Loss and Accuracy")  
plt.xlabel("Epoch #")  
plt.ylabel("Loss/Accuracy")  
plt.legend()

Out [23]: <matplotlib.legend.Legend at 0x1654d1fe50>



In [ ]: