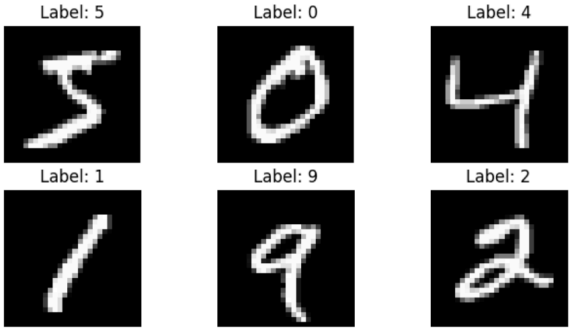
EXPERIMENT 4

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Training data shape: (60000, 28, 28)
Testing data shape: (10000, 28, 28)
```




```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/reshaping/flatten.py:37: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When us
  super().__init__(**kwargs)
Model: "sequential"
```

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 784)	0
dense (Dense)	(None, 128)	100,480
dense_1 (Dense)	(None, 10)	1,290

```
 Total params: 101,770 (397.54 KB)
 Trainable params: 101,770 (397.54 KB)
 Non-trainable params: 0 (0.00 B)
Epoch 1/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8737 - loss: 0.4564 - val_accuracy: 0.9647 - val_loss: 0.1269



 Trainable params: 101,770 (397.54 KB)
 Non-trainable params: 0 (0.00 B)
Epoch 1/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8737 - loss: 0.4564 - val_accuracy: 0.9647 - val_loss: 0.1269
Epoch 2/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 8s 4ms/step - accuracy: 0.9631 - loss: 0.1306 - val_accuracy: 0.9703 - val_loss: 0.1051
Epoch 3/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.9733 - loss: 0.0859 - val_accuracy: 0.9763 - val_loss: 0.0903
Epoch 4/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9815 - loss: 0.0625 - val_accuracy: 0.9788 - val_loss: 0.0765
Epoch 5/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.9851 - loss: 0.0485 - val_accuracy: 0.9775 - val_loss: 0.0784
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.98      0.99      0.99      1135
           2       0.98      0.97      0.98      1032
           3       0.94      0.99      0.96      1010
           4       0.98      0.98      0.98       982
           5       0.99      0.95      0.97       892
           6       0.98      0.98      0.98       958
           7       0.98      0.97      0.98      1028
           8       0.96      0.98      0.97       974
           9       0.98      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.98      0.97      0.97     10000
weighted avg       0.98      0.97      0.97     10000

Confusion Matrix:
[[ 970    1    2    1    1    0    2    1    2    0]
 [   0 1127    3    0    0    0    2    0    3    0]
 [   2    4 1005    5    0    0    3    3   10    0]
 [   0    1    2  996    0    1    0    3    5    2]
 [   0    1    3    2  959    0    7    1    1    8]
 [   3    0    0   21    1  850    6    0    9    2]
 [   4    2    1    1    4    4  938    0    4    0]
 [   0    6    9    8    0    0    0  996    3    6]
 [   2    0    2    7    3    2    2    4  952    0]
```

```
           7       0.98      0.97      0.98      1028
           8       0.96      0.98      0.97       974
           9       0.98      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.98      0.97      0.97     10000
weighted avg       0.98      0.97      0.97     10000

Confusion Matrix:
[[ 970    1    2    1    1    0    2    1    2    0]
 [   0 1127    3    0    0    0    2    0    3    0]
 [   2    4 1005    5    0    0    3    3   10    0]
 [   0    1    2  996    0    1    0    3    5    2]
 [   0    1    3    2  959    0    7    1    1    8]
 [   3    0    0   21    1  850    6    0    9    2]
 [   4    2    1    1    4    4  938    0    4    0]
 [   0    6    9    8    0    0    0  996    3    6]
 [   2    0    2    7    3    2    2    4  952    0]
 [   5    3    2   16    9    5    1    5    6  957]]
```