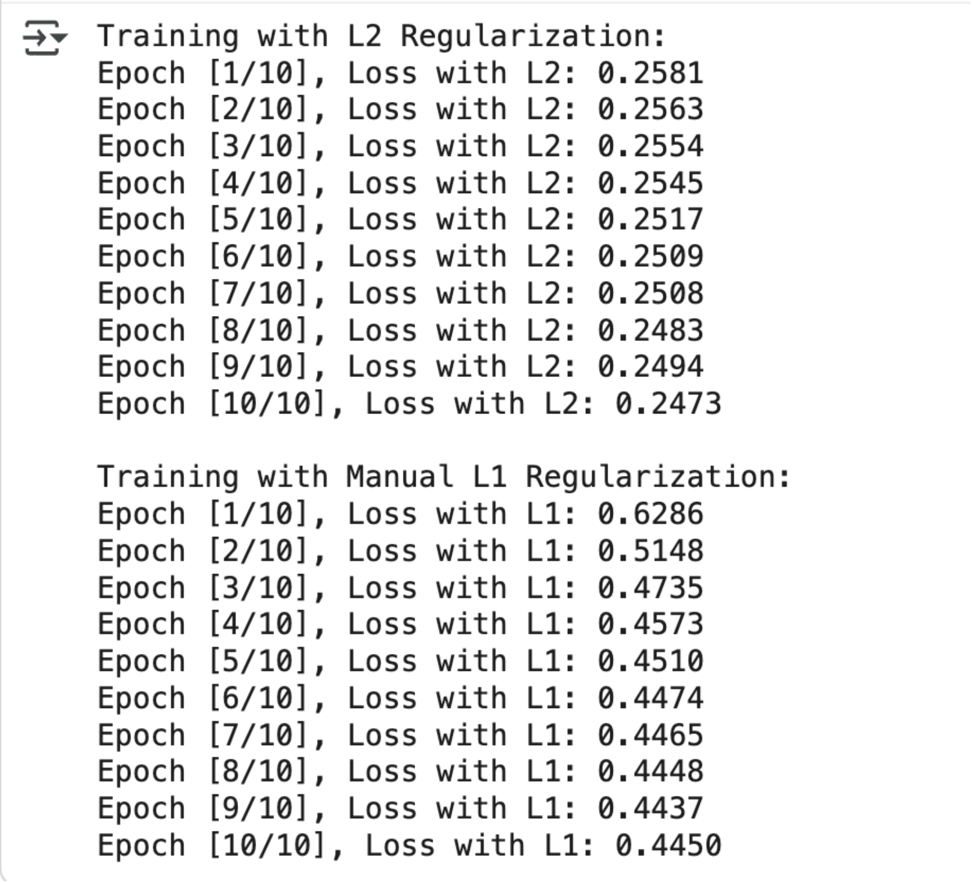
Assignment1 Report (Rosie Wang 1806394)

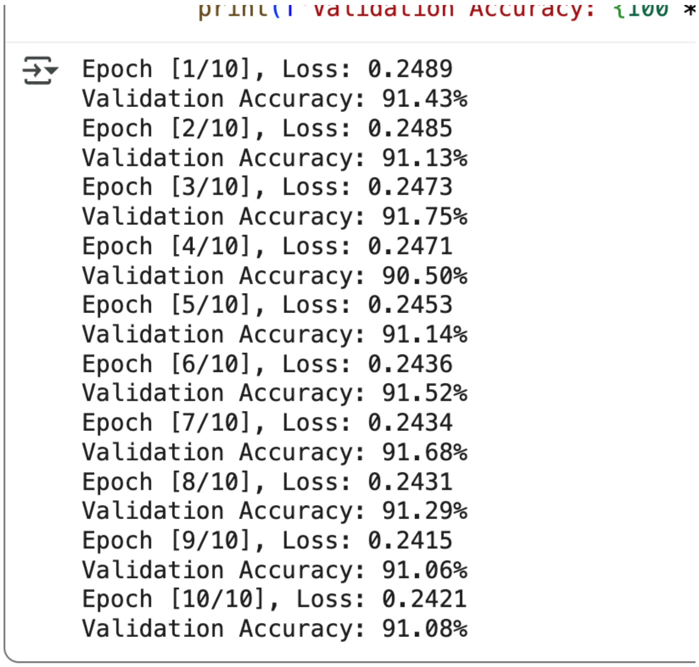
- ## Which regularization worked better (L1 vs L2)?

L2 works better in my code(L1_lambda=0.001) with less training loss in the same epoch.



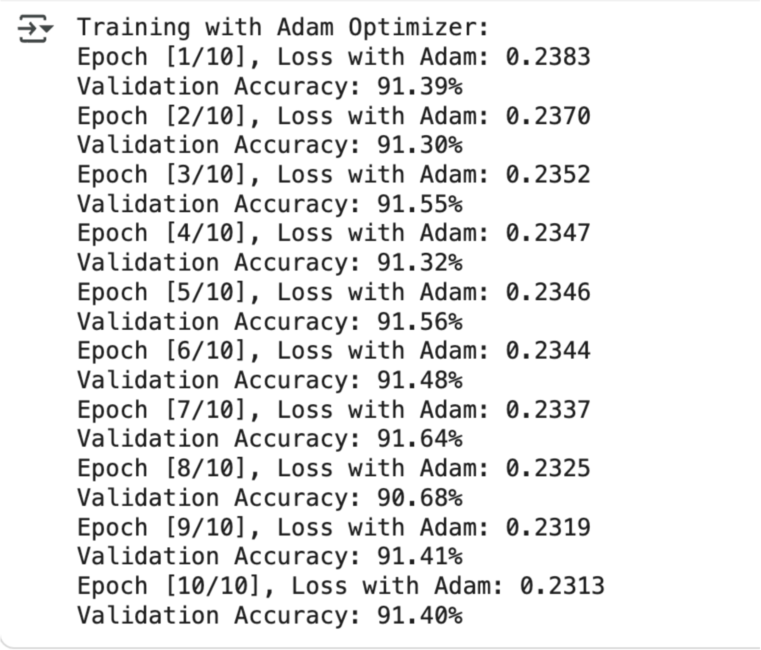
```
Training with L2 Regularization:
Epoch [1/10], Loss with L2: 0.2581
Epoch [2/10], Loss with L2: 0.2563
Epoch [3/10], Loss with L2: 0.2554
Epoch [4/10], Loss with L2: 0.2545
Epoch [5/10], Loss with L2: 0.2517
Epoch [6/10], Loss with L2: 0.2509
Epoch [7/10], Loss with L2: 0.2508
Epoch [8/10], Loss with L2: 0.2483
Epoch [9/10], Loss with L2: 0.2494
Epoch [10/10], Loss with L2: 0.2473

Training with Manual L1 Regularization:
Epoch [1/10], Loss with L1: 0.6286
Epoch [2/10], Loss with L1: 0.5148
Epoch [3/10], Loss with L1: 0.4735
Epoch [4/10], Loss with L1: 0.4573
Epoch [5/10], Loss with L1: 0.4510
Epoch [6/10], Loss with L1: 0.4474
Epoch [7/10], Loss with L1: 0.4465
Epoch [8/10], Loss with L1: 0.4448
Epoch [9/10], Loss with L1: 0.4437
Epoch [10/10], Loss with L1: 0.4450
```

- ## Which optimizer was more effective (SGD vs Adam)?



```
            print(f"Validation Accuracy: {100 *
```

```
Epoch [1/10], Loss: 0.2489
Validation Accuracy: 91.43%
Epoch [2/10], Loss: 0.2485
Validation Accuracy: 91.13%
Epoch [3/10], Loss: 0.2473
Validation Accuracy: 91.75%
Epoch [4/10], Loss: 0.2471
Validation Accuracy: 90.50%
Epoch [5/10], Loss: 0.2453
Validation Accuracy: 91.14%
Epoch [6/10], Loss: 0.2436
Validation Accuracy: 91.52%
Epoch [7/10], Loss: 0.2434
Validation Accuracy: 91.68%
Epoch [8/10], Loss: 0.2431
Validation Accuracy: 91.29%
Epoch [9/10], Loss: 0.2415
Validation Accuracy: 91.06%
Epoch [10/10], Loss: 0.2421
Validation Accuracy: 91.08%
```

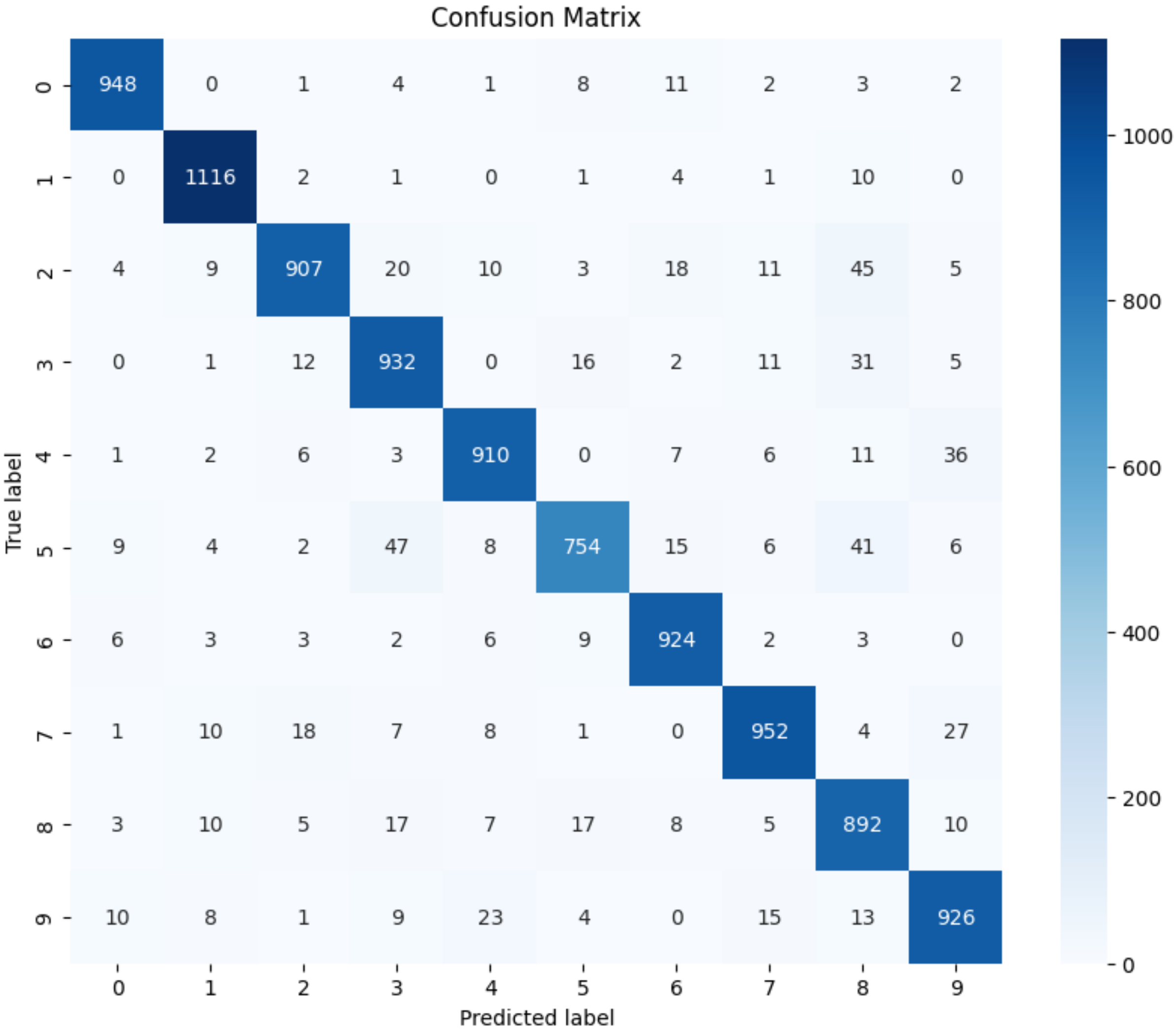


```
Training with Adam Optimizer:
Epoch [1/10], Loss with Adam: 0.2383
Validation Accuracy: 91.39%
Epoch [2/10], Loss with Adam: 0.2370
Validation Accuracy: 91.30%
Epoch [3/10], Loss with Adam: 0.2352
Validation Accuracy: 91.55%
Epoch [4/10], Loss with Adam: 0.2347
Validation Accuracy: 91.32%
Epoch [5/10], Loss with Adam: 0.2346
Validation Accuracy: 91.56%
Epoch [6/10], Loss with Adam: 0.2344
Validation Accuracy: 91.48%
Epoch [7/10], Loss with Adam: 0.2337
Validation Accuracy: 91.64%
Epoch [8/10], Loss with Adam: 0.2325
Validation Accuracy: 90.68%
Epoch [9/10], Loss with Adam: 0.2319
Validation Accuracy: 91.41%
Epoch [10/10], Loss with Adam: 0.2313
Validation Accuracy: 91.40%
```

According to the output, we can tell Adam (figure 2) has the higher validation accuracy compared to the SGD's validation accuracy in the same epoch. So Adam is more effective.

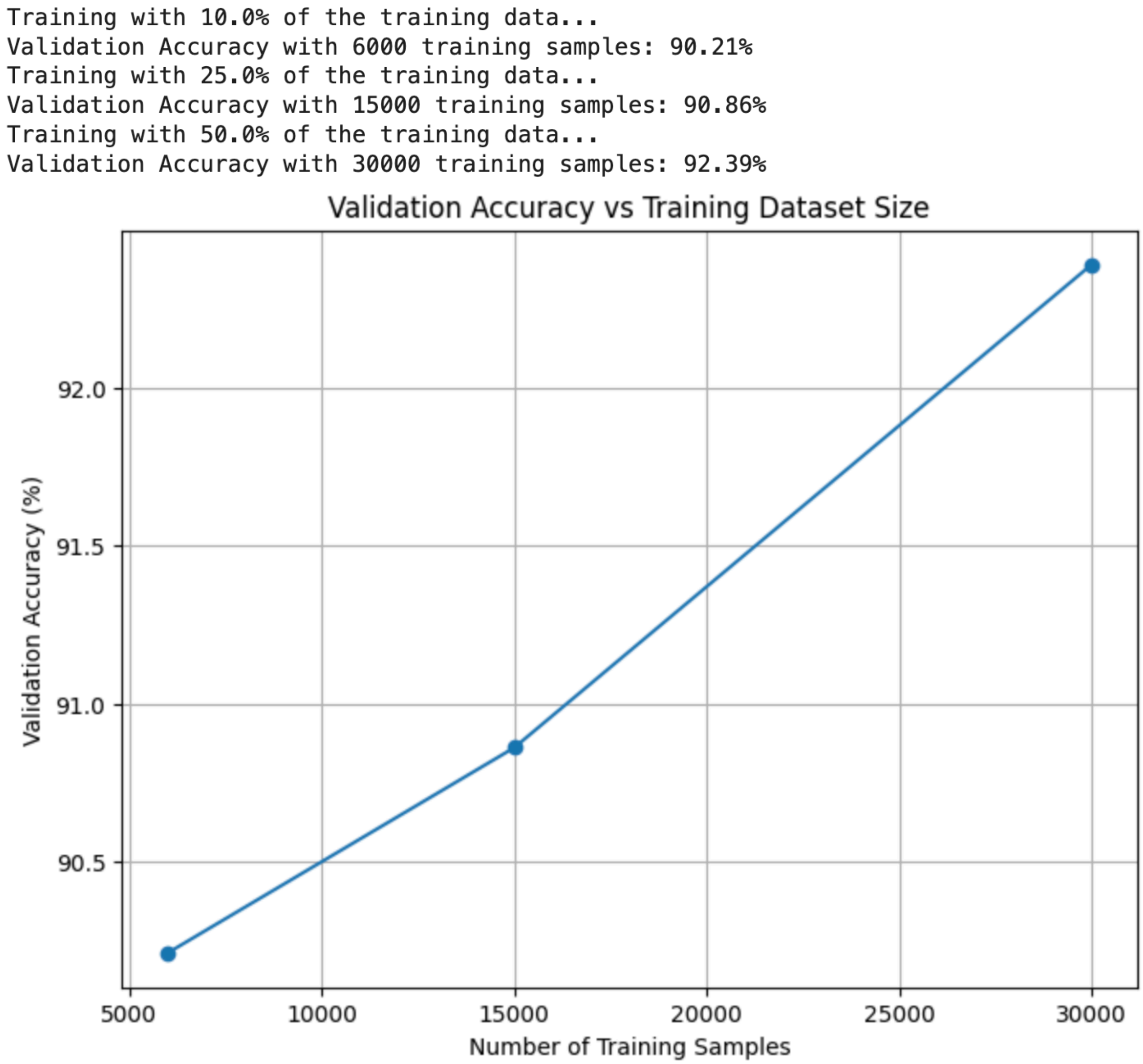
What does the confusion matrix reveal?

confusion matrix shows how is the performance the model hold. In my case, the confusion matrix reveals that logistic regression achieves high accuracy on simple digits like **0, 1, 7, and 9**, but struggles with visually similar pairs such as **2 vs 8, 3 vs 5/8, and 4 vs 9**. This highlights the model's limitation in capturing fine-grained local features compared to more advanced models like CNNs.



- ## How does performance degrade with reduced or noisy data?

Performance(from low to high): noisy data training < 10% training < 25% training < 50% training < 100% training.



```
Training with 10.0% of the training data...
Validation Accuracy with 6000 training samples: 90.21%
Training with 25.0% of the training data...
Validation Accuracy with 15000 training samples: 90.86%
Training with 50.0% of the training data...
Validation Accuracy with 30000 training samples: 92.39%
```

- **How did Gemini help (include sample prompts + modifications)?**

Gemini helped me understand the code and some specific and common syntax in pytorch and CNN. Also, it helps debug with the process of CNN and modifying with the hyperparameter etc.