**CS539 Assignment 04**

**Part 2**

When the Network only has two fully connected network, that is **without CNN** layer, we got the results as follows:

|  |
| --- |
| A screenshot of a computer  AI-generated content may be incorrect. |

Average Loss: 0.0990

Accuracy: 9693/10000 (97%)

When the network has 1 CNN layer with pooling and two FC layers, that is **with CNN layer**, it resulted as:

|  |
| --- |
|  |

Average Loss: 0.1585

Accuracy: 9554/10000 (96%)

**Experimenting with 3 alternative network topologies and hyper-parameters as:**

**#1. 2 CNN layers + 2 FC layers**

I used 2 CNN layers with pooling and 2 fully – connected (FC) layers.

|  |
| --- |
|  |

Average Loss: 0.1690

Accuracy: 9494/10000 (95%)

**#2. Learning Rate Tuning with 1 CNN layer model**

I tested with 3 different learning rates with a simplified model (1 CNN layer):

|  |  |  |  |
| --- | --- | --- | --- |
| **Learning Rate** | **Test Loss (Avg Loss)** | **Test Accuracy (in %)** | **Notes** |
| 0.01 | 0.0585 | 97.90% | Best balance of speed/accuracy |
| 0.001 | 0.1351 | 95.34% | Too slow convergence |
| 0.5 | 2.3293 | 10.28% | Diverged |

Best LR = 0.01: Achieved 97.9 % accuracy, the highest among the tested LRs.

**#3. Batch Size Tuning**

I tested with 3 different batch sizes with the model having 1 CNN layer with pooling and 2 FC layers with LR = 0.01 and momentum = 0.9

|  |  |  |  |
| --- | --- | --- | --- |
| **Batch Size** | **Test Loss (avg. loss)** | **Test Accuracy** | **Notes** |
| 16 | 0.0532 | 98.31% | Highest accuracy |
| 32 | 0.0581 | 98.03% | Default; stable |
| 64 | 0.0723 | 97.63% | Slightly higher loss |

**The best configuration was:**

1 CNN + 2 FC with the Test accuracy as 98.31% having LR = 0.01 and batch size = 16

Through me experimenting, I learned that **hyperparameter tuning significantly impacts model performance**. First, the **learning rate** must be carefully chosen—too high (0.5) causes divergence, while too low (0.001) slows convergence, with 0.01 proving optimal for stable training. Second, **batch size** affects both accuracy and training dynamics: smaller batches (16) yielded the highest accuracy (98.31%) due to noisier but more exploratory updates, while larger batches (64) traded slight accuracy loss (97.63%) for faster training. Finally, **model complexity** isn’t always better—a simpler 1-CNN-layer model outperformed a deeper 2-CNN-layer one (98.31% vs. 96.12%), suggesting that overly complex architectures can overfit even on simple datasets like MNIST. These results highlight the importance of **balanced hyperparameters**, **trade-offs between speed and precision**, and the **risk of over-engineering** models.