CS 599: Deep Learning - Lab 2

Neha Nagaraja
School of Informatics, Computing, and Cyber Systems
Northern Arizona University
nn454@nau.edu

1 Problem 1: Measuring Catastrophic Forgetting in Multi-layer perceptron

1.1 Objective

The primary goal of this experiment is to measure and analyze catastrophic forgetting in neural networks, specifically in multi-layer perceptrons (MLPs).

Catastrophic forgetting occurs when a neural network forgets previously learned information upon learning new tasks. This is a critical challenge in continual learning, where models need to retain knowledge while incrementally learning new tasks.

To study this problem, we use Permuted MNIST, a continual learning benchmark where the pixel positions of MNIST images are randomly shuffled for each task, making each task unique while keeping the labels intact.

1.2 Implementation

1.2.1 Step 1: Data Preparation

- Loaded the MNIST dataset
- Normalized and reshaped images from 28×28 to 784 pixels.
- Generated 10 unique pixel permutations (one for each task).
- Applied permutations to create distinct task versions of MNIST.
- Set a random seed from ”Neha” to ensure reproducibility.

Figure 1 is an example of original image and it’s permutating images.

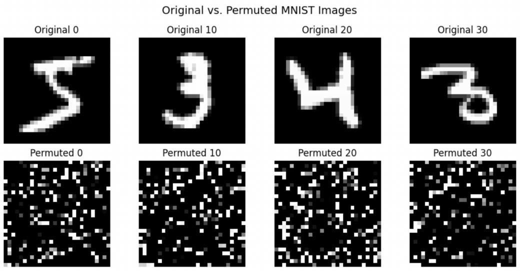

Figure 1: Original vs. Permuted MNIST Images

1.2.2 Train on Task A (Baseline Model)

We trained an MLP model on the first permuted MNIST task (Task A).

Preprint. Under review.

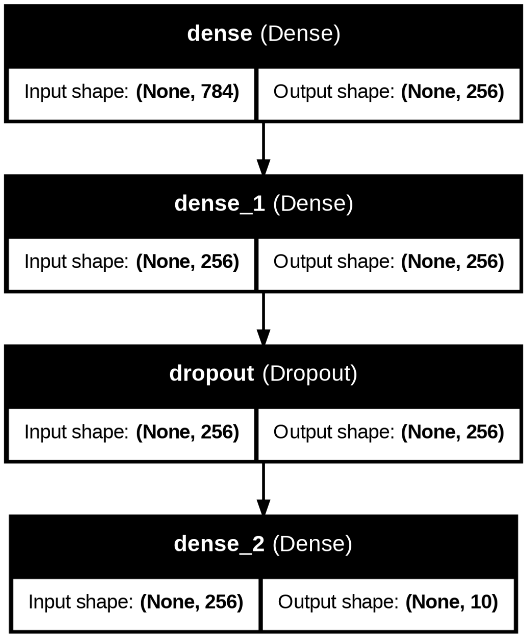

Figure 2: MLP model architecture

The model configuration:

- Depth: 2 hidden layers
- Dropout: 0.5
- Optimizer: Adam
- Loss Function: NLL

1.3 Architecture of of the MLP Model

Figure 2 shows the MLP Model Architecture

The model reached 97.88% validation accuracy on Task A after 50 epochs with a smooth decline in loss, indicating effective feature extraction and convergence. Minimal overfitting was observed, as the test accuracy remained high, suggesting robust generalization to unseen data.

1.3.1 Training on other Tasks

The accuracy on early tasks (Task 1, Task 2, etc.) declines as new tasks are introduced. This suggests that the model forgets previously learned tasks due to the lack of memory consolidation or regularization.

1.3.2 Compute Forgetting Metrics

The model's ACC (0.7269) suggests moderate overall performance across tasks, but the negative BWT (-0.2639) and CBWT (-0.1983) indicate significant forgetting of earlier tasks, confirming catastrophic forgetting. The positive TBWT (0.1791) suggests some transfer learning benefits, but they are insufficient to counterbalance the forgetting, highlighting the need for continual learning strategies such as experience replay or regularization techniques.

1.4 Effect of Loss functions

Model parameters: - Depth: 2, Dropout: 0.5, Optimizer: Adam, Loss Function: NLL/L1/L2,L1+L2.

Loss Functions Used

We experimented with different loss functions to analyze their impact on catastrophic forgetting in sequential learning. The loss functions used are:

Negative Log-Likelihood (NLL) The Negative Log-Likelihood loss is commonly used for classification tasks and is defined as:

$$\mathcal{L}_{NLL} = -\sum_{i=1}^{N} y_i \log \hat{y}_i \tag{1}$$

where y_i is the true label, $\hat{y}_i$ is the predicted probability, and N is the number of samples.

L1 Loss (Mean Absolute Error) L1 loss encourages sparsity by minimizing the absolute difference between predictions and true values:

$$\mathcal{L}_{L1} = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{2}$$

L2 Loss (Mean Squared Error) L2 loss penalizes large errors more than L1 by minimizing the squared difference:

$$\mathcal{L}_{L2} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{3}$$

Hybrid Loss (L1 + L2) A combination of L1 and L2 regularization helps balance sparsity and stability:

$$\mathcal{L}_{Hybrid} = \alpha \mathcal{L}_{L1} + (1 - \alpha) \mathcal{L}_{L2} \tag{4}$$

where α is a weighting factor controlling the contribution of each term.

The performance of these loss functions on forgetting metrics is summarized in Table 1.

- Hybrid Loss (L1 + L2) performed best, achieving the highest accuracy (0.7538) and the least forgetting (BWT: -0.2339, CBWT: -0.1780).
- L2 loss was a close second (ACC: 0.7346, BWT: -0.2560), showing better stability than L1.
- NLL outperformed L1, with less forgetting (BWT: -0.2639 vs. -0.2846).
- L1 loss showed the most forgetting, with the lowest accuracy (0.7096) and highest BWT drop.
- Takeaway: Hybrid and L2 loss reduce forgetting better than NLL and L1.

Loss Function	**ACC**	**BWT**	**TBWT**	**CBWT**
NLL	0.7269	-0.2639	0.1791	-0.1983
L1	0.7096	-0.2846	0.1657	-0.2106
L2	0.7346	-0.2560	0.1764	-0.1788
Hybrid	0.7538	-0.2339	0.1975	-0.1780

Table 1: Forgetting Metrics for Different Loss Functions

Figure 3 shows how different loss functions affect ACC, BWT, TBWT, and CBWT."

1.5 Effect of Dropout

Model parameters: - Depth: 2, Dropout: 0/0.2/0.5/0.6, Optimizer: Adam, Loss Function: L1+L2.

I used 0, 0.2, 0.5, 0.6 values to test the impact of dropout on the model.

Does Dropout Help?

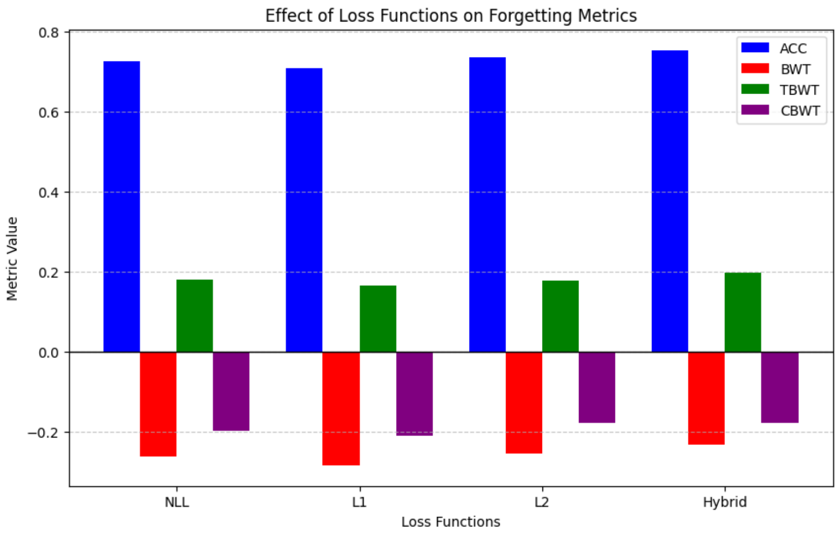

Figure 3: Effect of Loss Functions on Forgetting Metrics

- Yes, dropout helps mitigate catastrophic forgetting, but excessive dropout can hurt performance.
- Dropout = 0.5 gave the highest ACC (0.7538) and best overall metrics.
- No dropout (0.0) resulted in worse forgetting (BWT: -0.2979, TBWT: 0.1511), showing that dropout acts as a regularizer.
- However, too much dropout (0.6) led to even worse ACC (0.6897) and BWT (-0.3047), suggesting that excessive dropout reduces model capacity and learning ability.
- Optimal dropout (0.5) provides a good balance between preventing overfitting and retaining past knowledge.

Dropout Rate	**ACC**	**BWT**	**TBWT**	**CBWT**
0.5	0.7538	-0.2339	0.1975	-0.1780
0.0	0.6959	-0.2979	0.1511	-0.2095
0.2	0.6956	-0.2992	0.1429	-0.1960
0.6	0.6897	-0.3047	0.1481	-0.2158

Table 2: Impact of Dropout Rate on Forgetting Metrics

Table 2 shows the impact of different dropout rates on forgetting metrics, while Figure 4 visualizes these variations.

1.6 Effect of Depth

Model parameters: - Depth: 2/3/4, Dropout: 0.5, Optimizer: Adam, Loss Function: L1+L2.

I tried Depth = 2, 3, 4;

- More depth increases forgetting.
- Depth 2 had the highest ACC (0.7538) and best BWT (-0.2339), meaning it retained past knowledge better.
- Depth 3 started showing more forgetting (BWT: -0.2767, CBWT: -0.2110), affecting stability.
- Depth 4 performed the worst (ACC: 0.6732, BWT: -0.3233, CBWT: -0.2379), indicating that deeper networks struggle more in continual learning.

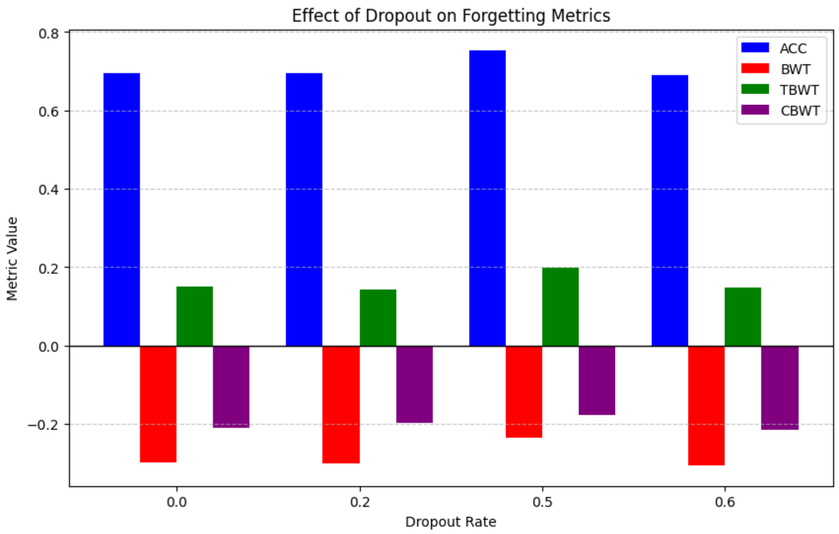

Figure 4: Effect of Dropout on Forgetting Metrics

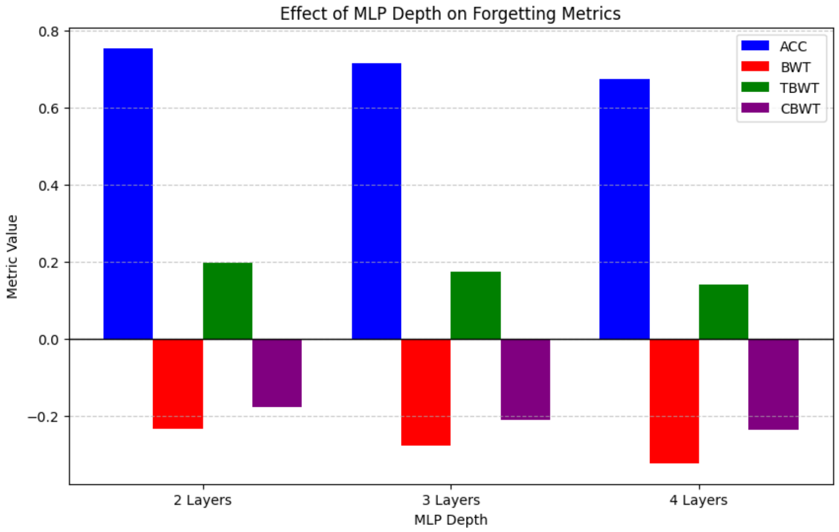

Figure 5: Effect of MLP Depth on Forgetting Metrics

- Shallow networks (depth=2) are more effective for minimizing catastrophic forgetting in this setting.

Table 3 presents the impact of model depth on forgetting metrics, while Figure 5 visualizes the performance trends across different depths.

Model Depth	ACC	BWT	TBWT	CBWT
2	0.7538	-0.2339	0.1975	-0.1780
3	0.7148	-0.2767	0.1731	-0.2110
4	0.6732	-0.3233	0.1414	-0.2379

Table 3: Impact of Model Depth on Forgetting Metrics

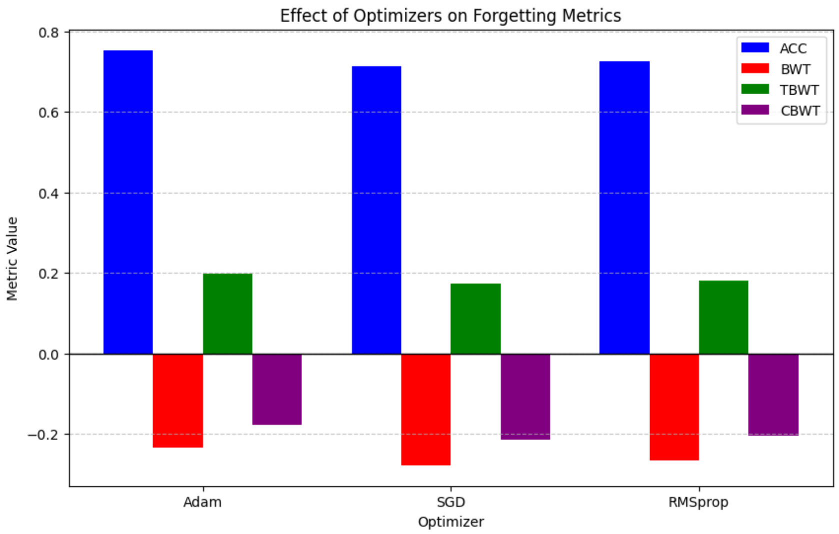

Figure 6: Effect of Optimizers on Forgetting Metrics

1.7 Effect of Optimizers

Model parameters: - Depth: 2, Dropout: 0.5, Optimizer: Adam/SGD/RMSProp, Loss Function: L1+L2.

I tried Adam, SGD & RMSProp.

Does the Optimizer Play a Role in Less Forgetting?

- Yes, optimizers significantly impact catastrophic forgetting.
- Adam performed the best (ACC: 0.7538, BWT: -0.2339, TBWT: 0.1975, CBWT: -0.1780), meaning it helped the model retain past tasks better.
- SGD had the worst BWT (-0.2775) and CBWT (-0.2138), showing higher forgetting.
- RMSprop performed in between Adam and SGD, but still had more forgetting than Adam.
- Adaptive optimizers (Adam) help the model retain old knowledge better, making them superior for continual learning tasks.

Table 4 shows the effect of different optimizers on forgetting metrics, while Figure 6 visualizes the performance differences.

Optimizer	**ACC**	**BWT**	**TBWT**	**CBWT**
Adam	0.7538	-0.2339	0.1975	-0.1780
SGD	0.7144	-0.2775	0.1738	-0.2138
RMSprop	0.7252	-0.2655	0.1804	-0.2046

Table 4: Impact of Optimizers on Forgetting Metrics

1.8 Validation Results plots

Figure 7 shows the validation loss over training, while Figure 8 presents the final accuracy for each task after sequential learning.

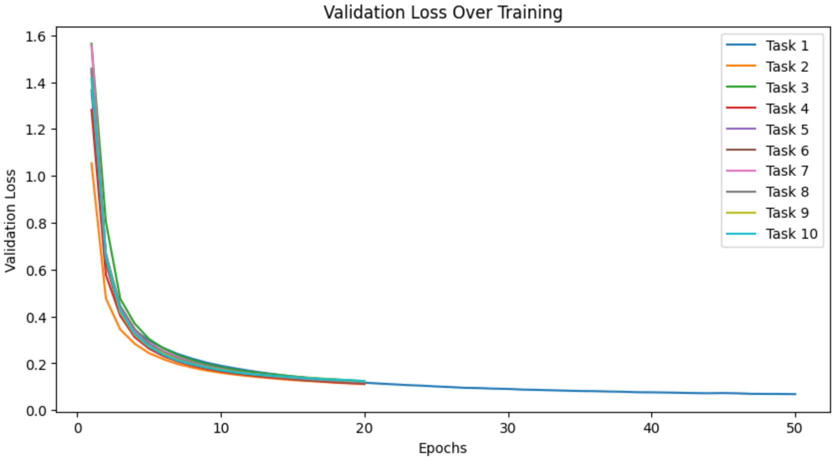

Figure 7: Validation Loss Over Training

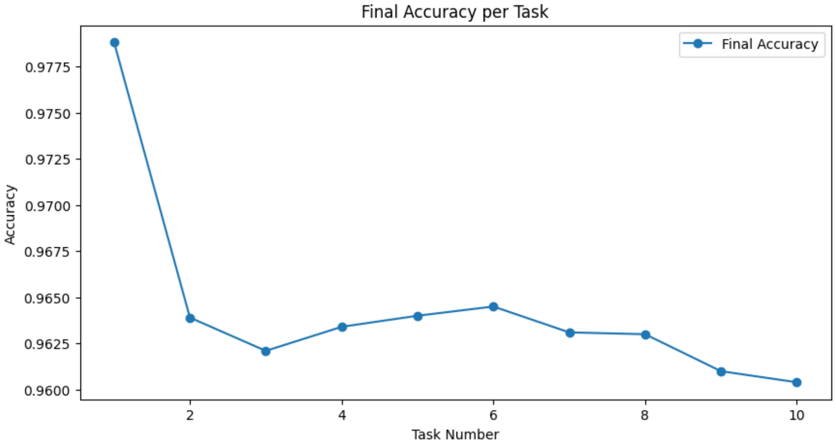

Figure 8: Final Accuracy per Task