Lab 2: Foundations of Deep Learning

Thi Hien Nguyen
School of Informatics, Computing, and Cyber Systems
Northern Arizona University
AZ, USA
tn598@nau.edu

The code can be accessed at `https://github.com/hienngt/IST597_Fall2019_TF2.0`.

1 Problem 1

In this report, I analyze catastrophic forgetting in neural networks using metrics such as average accuracy (ACC), backward transfer (BWT), temporal backward transfer (TBWT), and cumulative backward transfer (CBWT). Catastrophic forgetting occurs when a model forgets previously learned tasks while learning new ones. These metrics are computed using a task performance matrix R, where $R_{i,j}$ represents the accuracy on task j after training on task i. Additionally, I use a diagonal matrix G, where $G_{i,i}$ represents the performance on task i immediately after training it.

1. ACC: This metric calculates the average accuracy across all tasks after completing training on all tasks.
$$\text{ACC} = \frac{1}{T}\sum_{i=1}^{T} R_{T,i} \tag{1}$$
2. BWT: This metric measures the effect of learning new tasks on the performance of previously learned tasks.
$$\text{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} (R_{T,i} - R_{i,i}) \tag{2}$$
3. TBWT: This metric evaluates the change in performance on previous tasks after learning all tasks compared to their immediate post-training performance.
$$\text{TBWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} (R_{T,i} - G_{i,i}) \tag{3}$$
4. CBWT: This metric measures the cumulative effect of learning new tasks on the performance of a specific previous task t.
$$\text{CBWT(t)} = \frac{1}{T-t}\sum_{i=t+1}^{T} (R_{i,t} - R_{t,t}) \tag{4}$$

   Where:
   - T: Total number of tasks.
   - $R_{i,j}$: Accuracy on task j after training on task i.
   - $G_{i,i}$: Accuracy on task i immediately after training it.

In this experiment, 10 tasks were created by applying a unique random permutation to the pixels of the images for each task. This approach generates diverse variations of the original dataset, resulting in new tasks. An example of such pixel permutations is shown in Fig. 1.

Fig. 2 illustrates the architecture of the multi-layer perceptron (MLP) model used in this experiment.

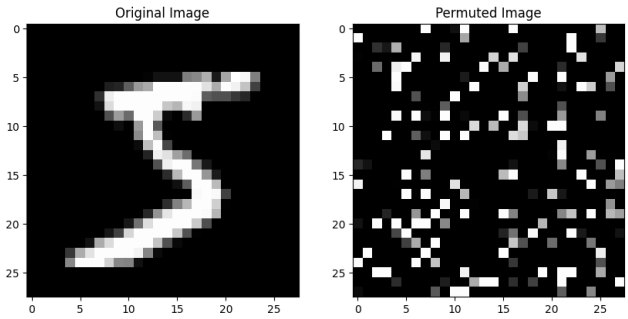

Figure 1: Permutation.

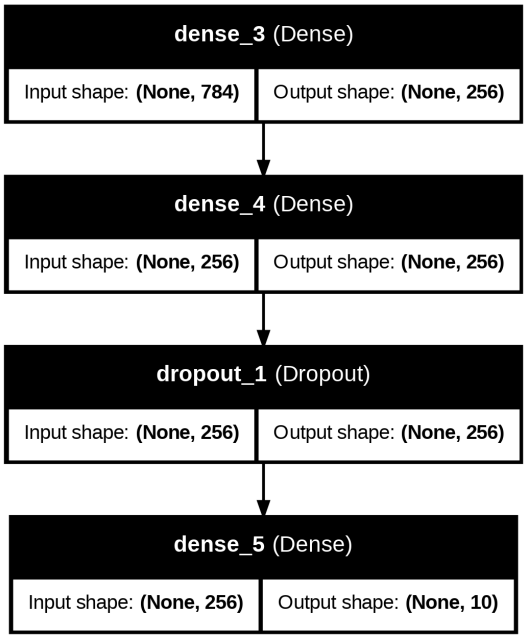

Figure 2: Model architecture.

1.1 Loss Functions

The following loss functions were used in the experiments:

1. Negative log-likelihood (NLL) loss
$$\text{NLL\_Loss}(\hat{y}) = -\log(\hat{y}) \tag{5}$$
2. L1 loss
$$\text{L1\_Loss}(y, \hat{y}) = |\hat{y} - y| \tag{6}$$
3. L2 loss
$$\text{L2\_Loss}(y, \hat{y}) = \sqrt{(\hat{y} - y)^2} \tag{7}$$
4. Hybrid loss (L1 + L2)
$$\text{Hybrid\_Loss}(y, \hat{y}) = L1\_Loss(y, \hat{y}) + L2\_Loss(y, \hat{y}) \tag{8}$$

These loss functions were utilized to optimize and evaluate the model's performance across various tasks.

Fig. 3 presents four matrices, each associated with different loss functions. It is evident that these loss functions significantly impact the model's capacity to remember information. Among them, the L1 loss function stands out for its effectiveness in reducing forgetting compared to the other loss functions employed in this model.

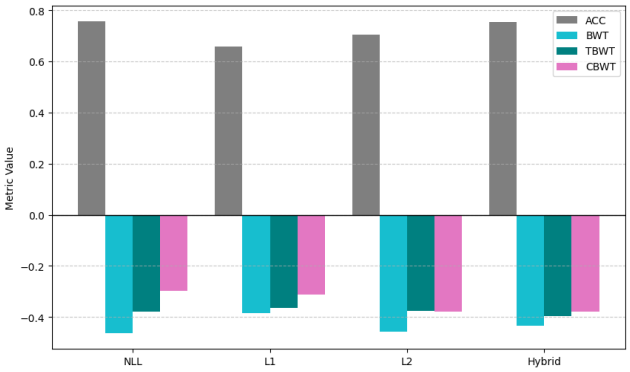

Figure 3: Loss function.

1.2 Dropout

I tried 4 dropout probablity $[0.0, 0.2, 0.3, 0.4]$. In Fig. 4, we can see that dropout helps mitigate catastrophic forgetting, with optimal performance observed at 0.3 and 0.4.

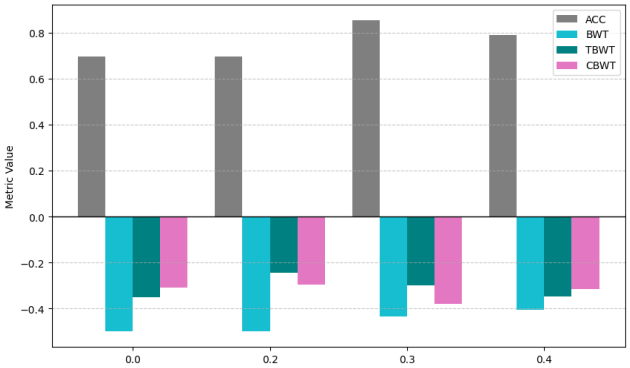

Figure 4: Dropout.

1.3 Depth

Fig. 5 shows that shallower networks achieve higher accuracy but suffer from more temporal and cumulative forgetting. Deeper networks mitigate temporal and cumulative forgetting but sacrifice accuracy.

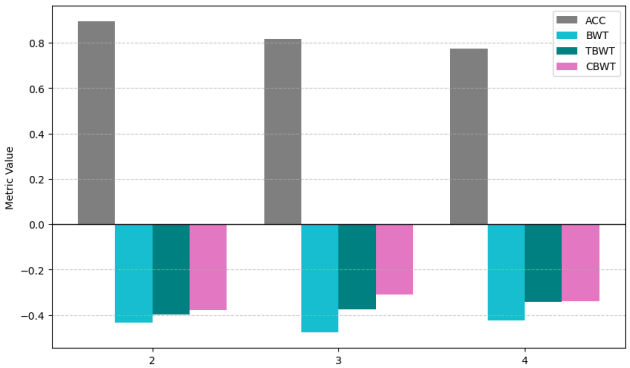

Figure 5: Depth.

1.4 Optimizers

Fig. 6 shows that Adam achieves the highest accuracy but suffers from significant forgetting (high negative BWT, TBWT, and CBWT). SGD performs better in terms of minimizing forgetting but sacrifices overall accuracy. RMSProp strikes a balance between accuracy and forgetting metrics.

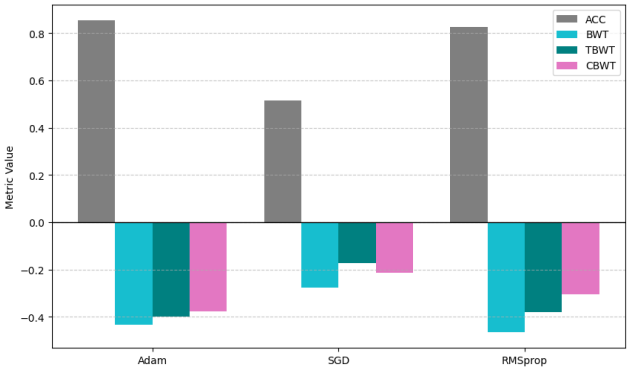

Figure 6: Optimizers.

1.5 Validation results

Fig. 7 illustrates the accuracy and loss of the original test dataset throughout the training of 10 tasks. As the number of epochs increases for other tasks, the model begins to forget information, resulting in a marked decline in performance on the original test dataset.

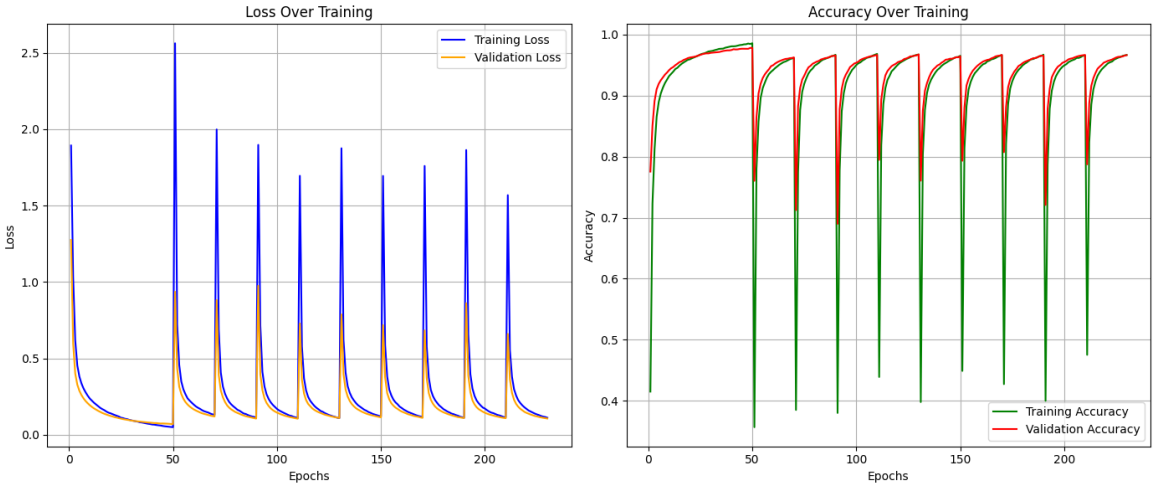

Figure 7: Validation results.