# DD2424 - Bonus Assignment 1

Silpa Soni Nallacheruvu

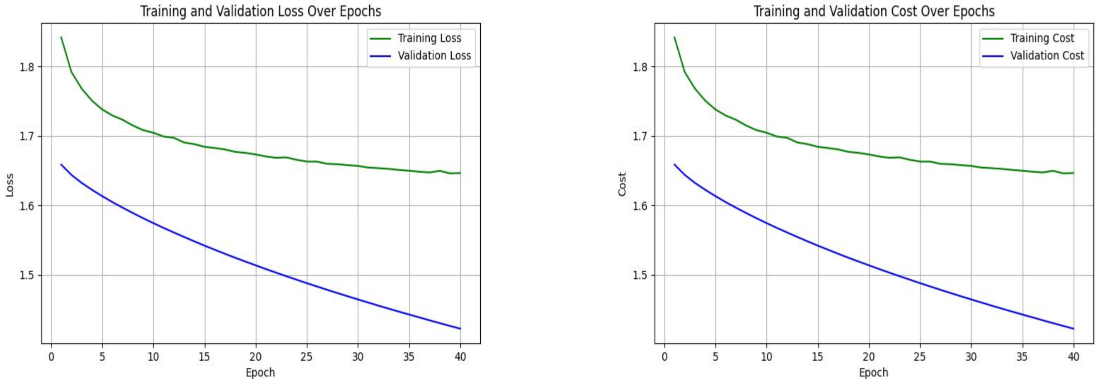
April 6, 2025

## Exercise 2.1

The goal of this question was to improve performance of the network.

### Q1.a

All the available training data was used for training and ran the classification again. Choosing the best parameter setting proven from the previous runs: n_batch=100, eta=.001, n_epochs=40 and lam=0.



(a) Training and Validation Loss

(b) Training and Validation Cost

Figure 1: Loss and cost plots for $\eta = 0.001$, $n\_batch = 100$, $n\_epochs = 40$, $\lambda = 0$

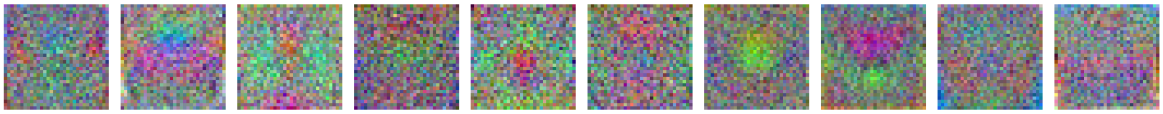


Figure 2: The learnt W matrix visualized with the settings: n_batch=100, eta=.001, n_epochs=40 and lam=0 for all training data.

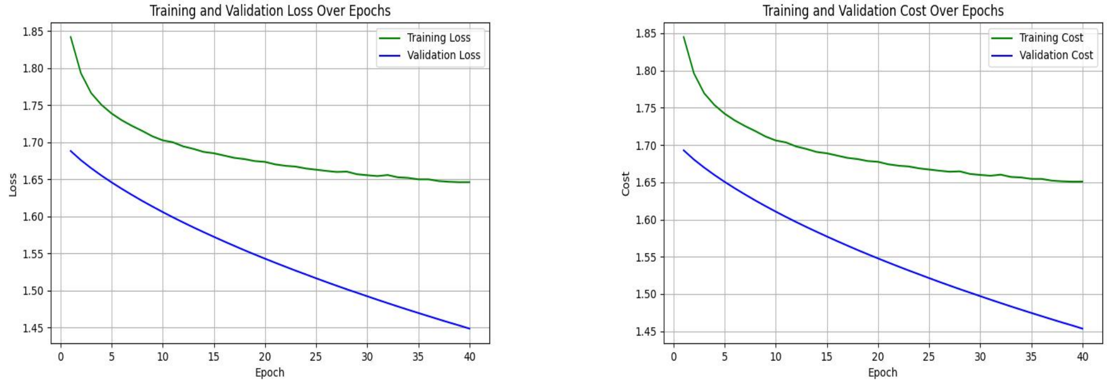
```
Accuracy for training : 44.42%
Accuracy for validation : 53.20%
Accuracy for testing : 47.24%
```

Listing 1: The accuracy for each run

With a larger training data and smaller learning rate, the filters begin to form smoother and more defined patterns, suggesting more stable and effective convergence without regularization. The validation loss and cost stay close to the training, but larger training data increased the cost and loss as well.

## Q1.b

All the available training data was augmented with a 0.5 probability and ran the classification again. Choosing the best parameter setting proven from the previous runs: n_batch=100, eta=.001, n_epochs=40 and lam=0.001 with a small regularization value $\lambda$.



(a) Training and Validation Loss

(b) Training and Validation Cost

Figure 3: Loss and cost plots for $\eta = 0.001$, $n\_batch = 100$, $n\_epochs = 40$, $\lambda = 0.001$

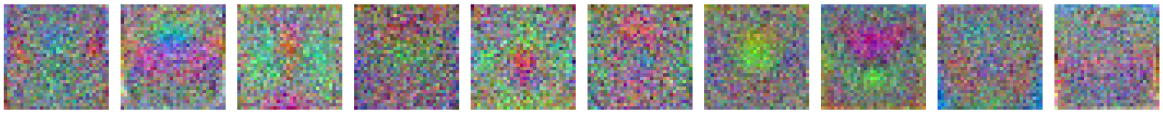


Figure 4: The learnt W matrix visualized with the settings: n_batch=100, eta=0.001, n_epochs=40 and lam=0.001 for all training data.

```
Accuracy for training : 44.33%
Accuracy for validation : 51.90%
Accuracy for testing : 47.25%
```

Listing 2: The accuracy for each run

By augmenting the data, the accuracy has remained the same. Although, the gap between training and validation performance in terms of loss and cost has reduced visibly.

## Q1.c

To optimize training, we perform a grid search over selected values of the learning rate ($\eta \in [0.001, 0.005, 0.01]$), L2 regularization strength ($\lambda \in [0, 0.001, 0.1, 1]$), and mini-batch size (n_batch $\in$ [50, 100, 200]). Each combination is evaluated using validation accuracy after 40 training epochs. The top 10 performing configurations are printed to identify the best parameter set for final training.

```
    Lambda    Eta  BatchSize  ValAccuracy
10   0.001  0.001        100         40.8
0    0.000  0.001         50         40.7
14   0.001  0.005        200         40.6
5    0.000  0.005        200         40.5
9    0.001  0.001         50         40.0
2    0.000  0.001        200         39.9
11   0.001  0.001        200         39.8
17   0.001  0.010        200         39.6
1    0.000  0.001        100         39.5
4    0.000  0.005        100         39.5
```

Listing 3: The top 10 validation accuracies in order of best to worst of each parameter setting

Hence, we can see that the best parameter setting is : n_batch=100, eta=0.001, n_epochs=40 and lam=0.001.

## Q1.d

To implement step decay for the learning rate, we will reduce it by a factor of 10 every 10 epochs for the best parameter setting from above: n_batch=100, eta=0.001, n_epochs=40 and lam=0.001.

```
Epoch 1/40 - LR: 0.00100 - Loss: 1.8440, Accuracy: 36.23%
Epoch 2/40 - LR: 0.00100 - Loss: 1.7923, Accuracy: 38.53%
Epoch 3/40 - LR: 0.00100 - Loss: 1.7676, Accuracy: 39.57%
Epoch 4/40 - LR: 0.00100 - Loss: 1.7514, Accuracy: 40.35%
Epoch 5/40 - LR: 0.00100 - Loss: 1.7389, Accuracy: 40.76%
Epoch 6/40 - LR: 0.00100 - Loss: 1.7282, Accuracy: 41.09%
Epoch 7/40 - LR: 0.00100 - Loss: 1.7220, Accuracy: 41.13%
Epoch 8/40 - LR: 0.00100 - Loss: 1.7132, Accuracy: 41.75%
Epoch 9/40 - LR: 0.00100 - Loss: 1.7120, Accuracy: 41.66%
Epoch 10/40 - LR: 0.00100 - Loss: 1.7029, Accuracy: 42.19%
Epoch 10: Reducing learning rate to 0.0001
Epoch 11/40 - LR: 0.00010 - Loss: 1.7009, Accuracy: 42.26%
Epoch 12/40 - LR: 0.00010 - Loss: 1.7003, Accuracy: 42.21%
Epoch 13/40 - LR: 0.00010 - Loss: 1.6997, Accuracy: 42.26%
Epoch 14/40 - LR: 0.00010 - Loss: 1.6992, Accuracy: 42.30%
Epoch 15/40 - LR: 0.00010 - Loss: 1.6987, Accuracy: 42.40%
Epoch 16/40 - LR: 0.00010 - Loss: 1.6983, Accuracy: 42.36%
Epoch 17/40 - LR: 0.00010 - Loss: 1.6979, Accuracy: 42.37%
Epoch 18/40 - LR: 0.00010 - Loss: 1.6974, Accuracy: 42.34%
Epoch 19/40 - LR: 0.00010 - Loss: 1.6970, Accuracy: 42.33%
Epoch 20/40 - LR: 0.00010 - Loss: 1.6966, Accuracy: 42.37%
Epoch 20: Reducing learning rate to 1e-05
Epoch 21/40 - LR: 0.00001 - Loss: 1.6965, Accuracy: 42.38%
Epoch 22/40 - LR: 0.00001 - Loss: 1.6964, Accuracy: 42.38%
Epoch 23/40 - LR: 0.00001 - Loss: 1.6964, Accuracy: 42.42%
Epoch 24/40 - LR: 0.00001 - Loss: 1.6963, Accuracy: 42.43%
Epoch 25/40 - LR: 0.00001 - Loss: 1.6963, Accuracy: 42.45%
Epoch 26/40 - LR: 0.00001 - Loss: 1.6962, Accuracy: 42.45%
Epoch 27/40 - LR: 0.00001 - Loss: 1.6962, Accuracy: 42.45%
Epoch 28/40 - LR: 0.00001 - Loss: 1.6961, Accuracy: 42.46%
Epoch 29/40 - LR: 0.00001 - Loss: 1.6961, Accuracy: 42.46%
Epoch 30/40 - LR: 0.00001 - Loss: 1.6961, Accuracy: 42.46%
Epoch 30: Reducing learning rate to 1.0000000000000002e-06
Epoch 31/40 - LR: 0.00000 - Loss: 1.6961, Accuracy: 42.46%
Epoch 32/40 - LR: 0.00000 - Loss: 1.6961, Accuracy: 42.46%
Epoch 33/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 34/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 35/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 36/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 37/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 38/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 39/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
Epoch 40/40 - LR: 0.00000 - Loss: 1.6960, Accuracy: 42.46%
```

Listing 4: Learning rate decay with validation accuracies

As we can see the validation seems to plateau towards the end after 30 epochs. This suggests that the model has mostly converged by around epoch 30. Further training yields minimal improvement, indicating it may be a good point to decay the learning rate or stop early.

Among all four strategies, using data augmentation combined with a small regularization value (Q1.b) gave the best generalization. Although the highest test accuracy (47.25%) was marginally higher than the others, what stands out is the smaller gap between training and validation loss, showing reduced overfitting. Step decay (Q1.d) helped stabilize the training, especially after the 30th epoch when the learning rate was reduced, but it didn't significantly improve test accuracy beyond what was already achieved. Grid search (Q1.c) helped identify optimal parameters, but the best gains were realized only when combined with augmentation. Using the full dataset without augmentation (Q1.a) improved test performance compared to initial smaller-batch experiments, but lacked the regularization advantages of augmentation.

## Exercise 2.2

The goal of this question is to train network - multiple binary cross-entropy losses

Given the multiple binary cross-entropy loss defined as:

$$\ell_{\text{multiple bce}}(\mathbf{x}, \mathbf{y}) = -\frac{1}{K} \sum_{k=1}^{K} \left[ (1 - y_k) \log(1 - p_k) + y_k \log(p_k) \right]$$

where $\mathbf{p} = \sigma(\mathbf{s})$ is the sigmoid activation applied element-wise to the input scores $\mathbf{s}$, our goal is to compute the gradient of the loss with respect to $\mathbf{s}$.

We first recall the derivative of the sigmoid function:

$$\frac{\partial p_k}{\partial s_k} = \frac{d}{ds_k} \left( \frac{1}{1 + e^{-s_k}} \right) = \frac{e^{-s_k}}{(1 + e^{-s_k})^2} = \frac{1}{(1 + e^{-s_k})} \cdot \left( 1 - \frac{1}{1 + e^{-s_k}} \right) = p_k(1 - p_k)$$

Using the chain rule:

$$\frac{\partial \ell}{\partial s_k} = \frac{\partial \ell}{\partial p_k} \cdot \frac{\partial p_k}{\partial s_k}$$

We compute:

$$\frac{\partial \ell}{\partial p_k} = \frac{1}{K} \left( \frac{y_k}{p_k} - \frac{1 - y_k}{1 - p_k} \right)$$

Therefore:

$$\frac{\partial \ell}{\partial s_k} = \frac{1}{K}\left(\frac{y_k - p_k}{p_k(1-p_k)}\right) \cdot p_k(1-p_k) = \frac{1}{K}(p_k - y_k) = \frac{1}{K}(\sigma(\mathbf{s_k}) - y_k)$$

**Final expression:**

$$\frac{\partial \ell_{\text{multiple bce}}}{\partial \mathbf{s}} = \frac{1}{K}(\sigma(\mathbf{s}) - \mathbf{y})$$

This result shows that the gradient takes the same form as in softmax with cross-entropy loss, except that the sigmoid function is applied element-wise and each class is treated independently.

## Cross-Entropy loss and Multiple binary cross-entropy loss comparison

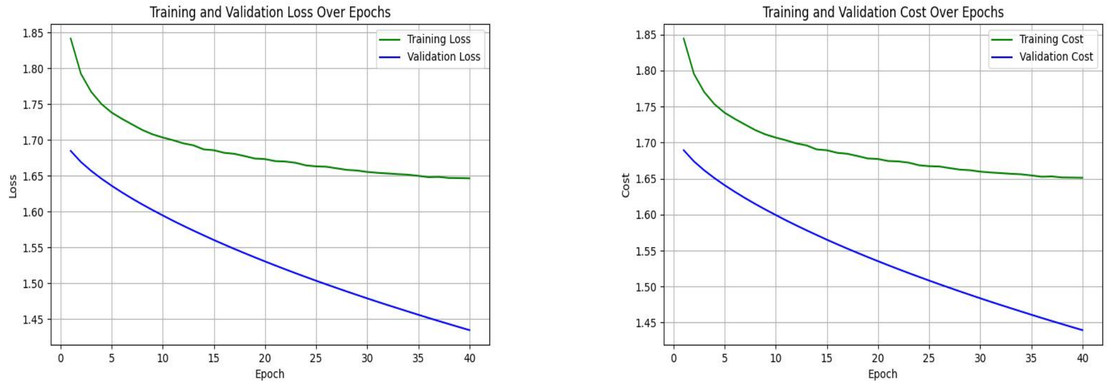
All the available training data is being used for the below calculations.

The final accuracies after using Cross entropy loss function and the following original parameter settings: n_batch=100, eta=0.001, n_epochs=40 and lam=0.001.

```
Accuracy for training : 44.37%
Accuracy for validation : 53.70%
Accuracy for testing : 46.92%
```

Listing 5: The final training, validation and test accuracies for CE loss

The following are the Cost and CE Loss plots for the same:



(a) Training and Validation CE Loss

(b) Training and Validation CE Cost

Figure 5: CE Loss and cost plots for $\eta = 0.001$, $n\_batch = 100$, $n\_epochs = 40$, $\lambda = 0.001$

Here is the histogram plot of the probability for the ground truth class for the examples correctly and those incorrectly classified using softmax + cross-entropy training.

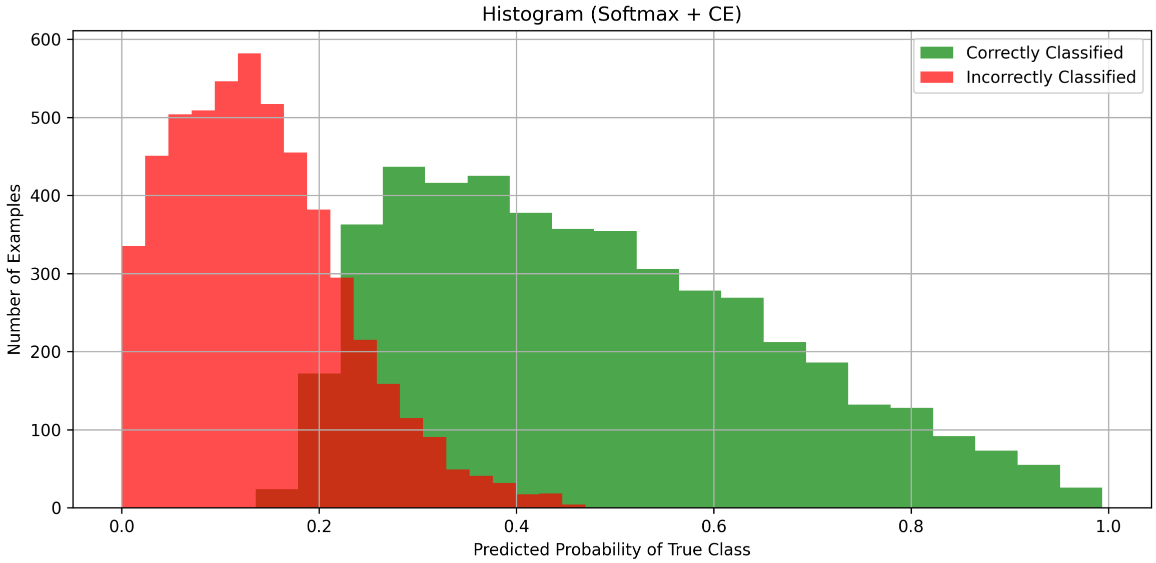


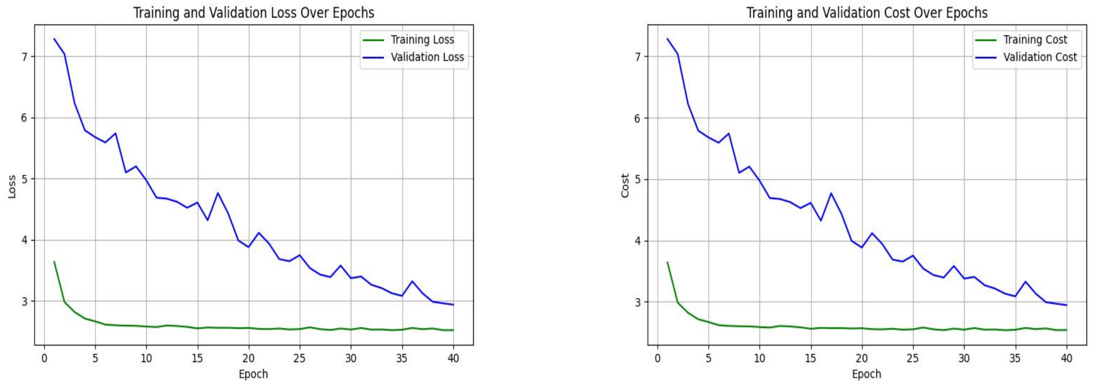
Figure 6: The histogram plot with the settings: n_batch=100, eta=.001, n_epochs=40 and lam=0.001 for all training data using CE loss.

Now, let us look at the final accuracies after using BCE loss function and the following original parameter settings: n_batch=100, eta=0.01, n_epochs=40 and lam=0.001. Here $\eta$ is adjusted to 0.01 from 0.001 to compensate for the K=10 factor in BCE loss.

```
Accuracy for training : 45.18%
Accuracy for validation : 79.00%
Accuracy for testing : 51.87%
```

Listing 6: The final training, validation and test accuracies for BCE loss

The following are the Cost and BCE Loss plots for the same:



(a) Training and Validation BCE Loss

(b) Training and Validation BCE Cost

Figure 7: BCE Loss and cost plots for $\eta = 0.01$, $n\_batch = 100$, $n\_epochs = 40$, $\lambda = 0.001$

Here is the histogram plot of the probability for the ground truth class for the examples correctly and those incorrectly classified using sigmoid + mul- tiple binary cross-entropy training.

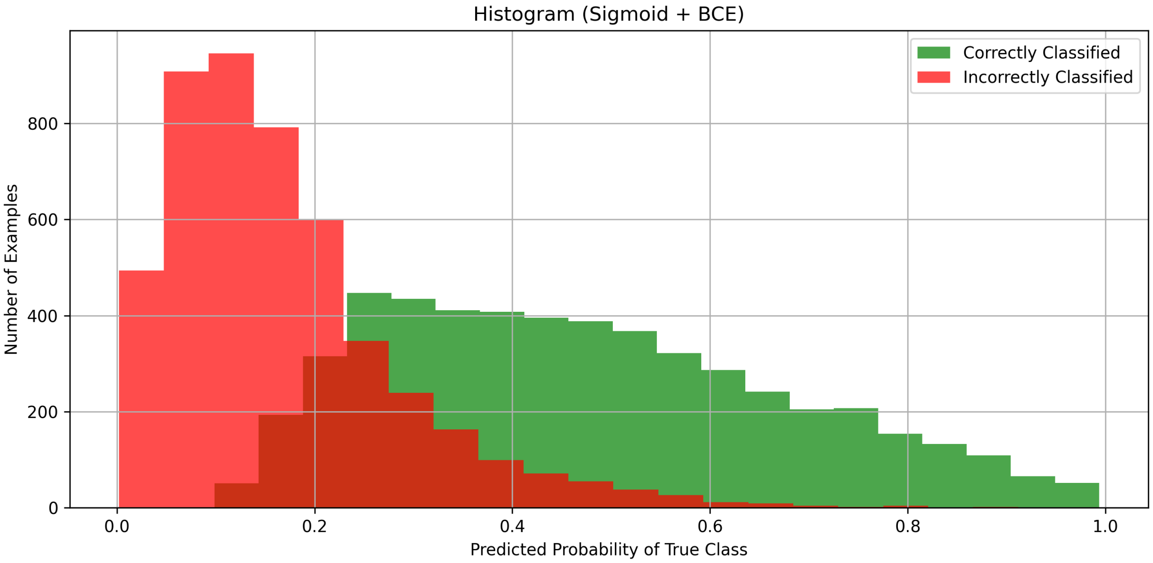


Figure 8: The histogram plot with the settings: n_batch=100, eta=.01, n_epochs=40 and lam=0.001 for all training data using BCE loss.

The testing and validation accuracies have definitely improved but the cost and loss for training and validation has increased by using BCE instead of CE.

The histogram plots show a qualitative difference between CE and BCE training. With softmax + CE, the model makes more confident predictions (higher probabilities) for both correct and incorrect classifications. In contrast, the BCE-trained model exhibits less confidence on incorrect predictions, spreading out the probability mass more evenly. This suggests improved calibration of the model under BCE, as the model is less overconfident when it is wrong.

### Overfitting Conclusion for BCE Vs CE

Training with BCE leads to higher validation and test accuracies, but also increased training loss and cost, indicating a greater ability to generalize while still fitting the training data well. Despite the increased loss, the validation performance improves, showing less overfitting compared to softmax + CE. Thus, BCE with an adjusted learning rate appears to reduce overfitting more effectively in this case.