İhsan Doğramacı Bilkent University

CS-464

Homework 2

Tuğberk Dikmen

21802480

Question 1)

1.1)

```
PVE for the first 10 principal components: 0.4881
```

High Variance Capture is capturing nearly half of the variance in the dataset with just 10 components out of 784 is significant. This indicates that these components are capturing essential features of the data.

This result is useful for dimensionality reduction, where we aim to reduce the number of features while retaining most of the information. In machine learning tasks such as image classification, reducing dimensions can lead to more efficient models without a substantial loss in performance.

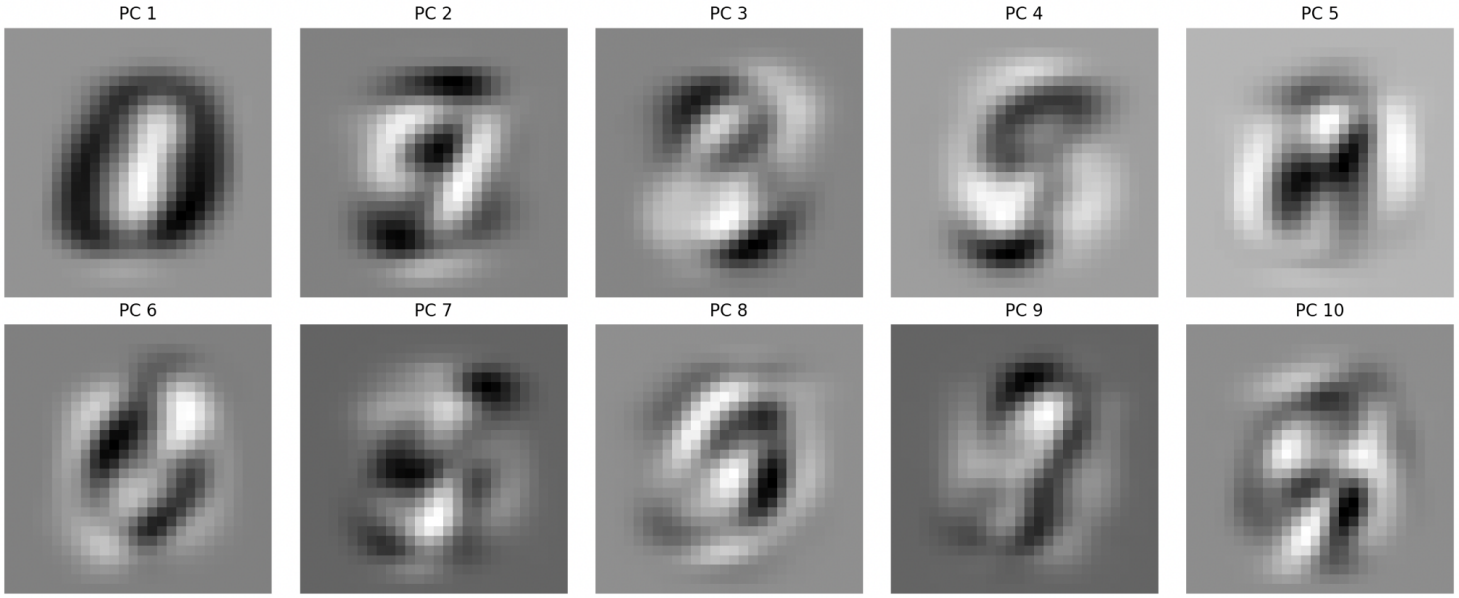
Each of these principal components represents a direction in the feature space along which the data varies the most. In the context of the MNIST dataset, these could be capturing fundamental stroke patterns, contrasts, or shapes common across different digits.

1.2)

```
Number of components to explain at least 70% of the variance: 26
```

To explain at least 70% of the variance in the data, a minimum of 26 principal components should be used. This indicates that selecting the top 26 components out of the total 784 will capture the majority of the information (70%) present in the dataset.

1.3)



These principal components are essentially capturing the most significant variations in the dataset. Each image might be thought of as a pattern or a feature that, when combined, can represent the digit images in the dataset.

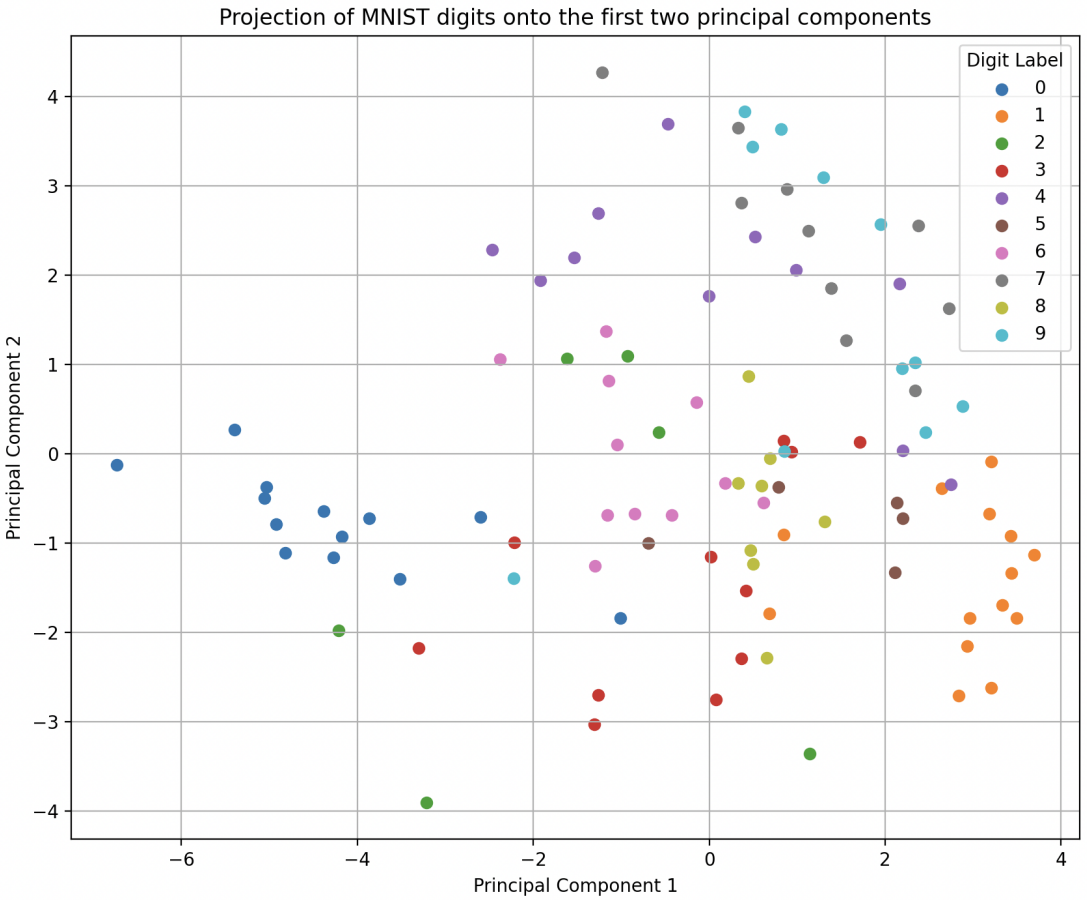
You might notice certain patterns or contrasts in these images. Some might represent common strokes or edges in digit images, like curves, lines, or intersections. For example, some components might be capturing vertical or horizontal edges, while others might be capturing curves or corners typical in digits.

These components are abstract representations and are not as straightforward to interpret as the original images. They are mathematical constructs that best describe the data variance and are not necessarily intuitive visual patterns.

These components can be used to reconstruct the original images. By combining these components with their respective weights, we can approximate the original digit images, with the first few components capturing the most critical aspects of the data.

Each principal component captures different aspects of the data. The first few components generally capture the most prominent features, and as we move to higher components, they capture more refined and specific details.

1.4)



From the plot, you can observe how the images of different digits cluster in the 2-D space formed by the first two principal components. Some digit labels may form distinct clusters, while others might overlap significantly.

Relating this to the visuals of the first two principal components from Question 1.3:

Principal Component 1 (PC1): Likely captures variance in one set of features (e.g., certain types of edges, curves, or contrasts).

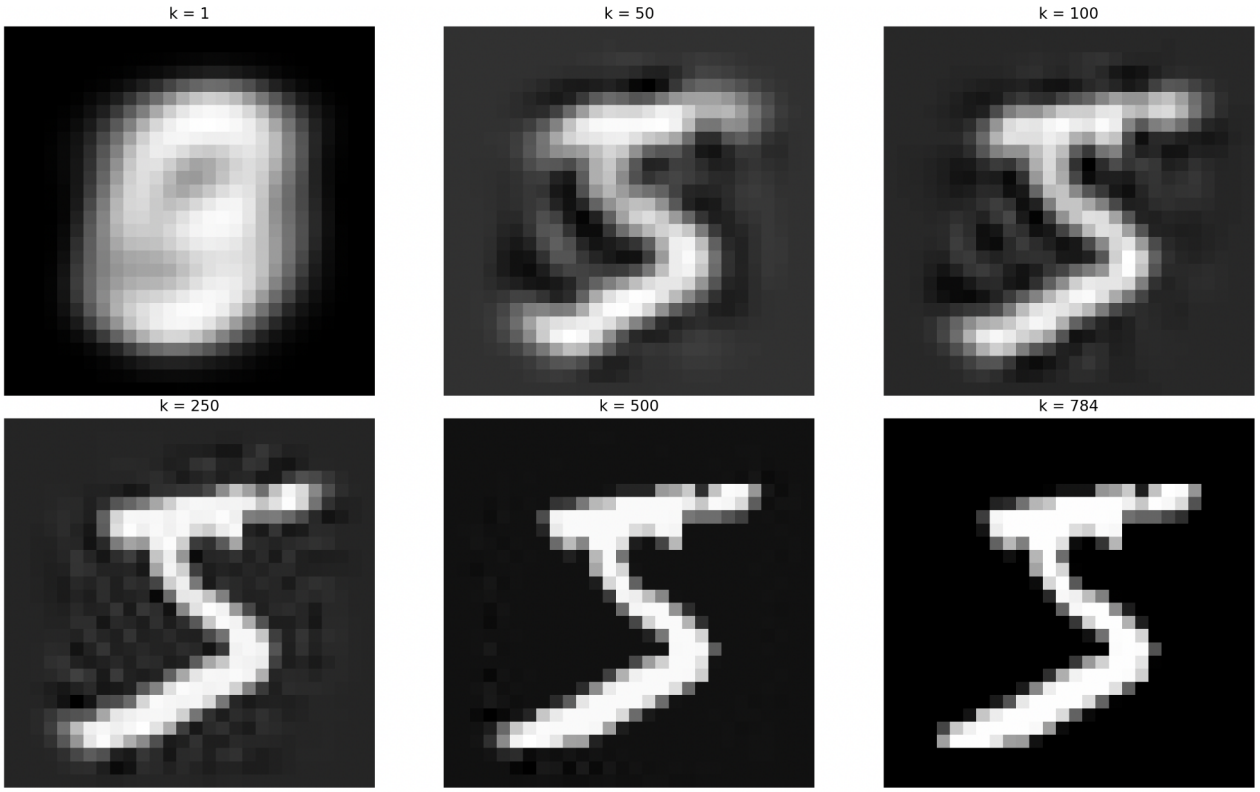
Principal Component 2 (PC2): Captures variance in a different set of features.

The way points are distributed indicates which features are more distinct among different digit classes. For example, if two digit labels are well separated, it suggests that the features captured by the first two principal components are quite distinct for those digits.

Overlapping clusters suggest that for some digits, the features captured by the first two PCs are similar. Conversely, well-separated clusters imply a distinctiveness in these features for those digits.

This visualization provide insights into the complexity of classifying digits in the MNIST dataset. Digits that form distinct clusters might be easier to classify than those which overlap significantly.

1.5)



k = 1: With only one principal component, the reconstructed image is extremely basic and lacks detailed features. It captures only the most dominant variance in the image, resulting in a very blurry and abstract representation.

Increasing k (50, 100, 250): As k increases, more details are captured in the reconstruction. The images become progressively clearer and more detailed. This shows how additional principal components capture more of the finer variations in the data.

Higher k (500, 784): At k=500 and especially k=784 (which includes all principal components), the reconstructed image closely resembles the original image. This is expected since using all principal components should theoretically allow for a perfect reconstruction, barring any numerical inaccuracies.

The results illustrate the trade-off in PCA between the number of components and the reconstruction quality. Fewer components lead to lower-dimensional representations (which is good for reducing complexity and computational costs) but at the cost of losing information. Conversely, using more components increases the reconstruction quality but also the complexity.

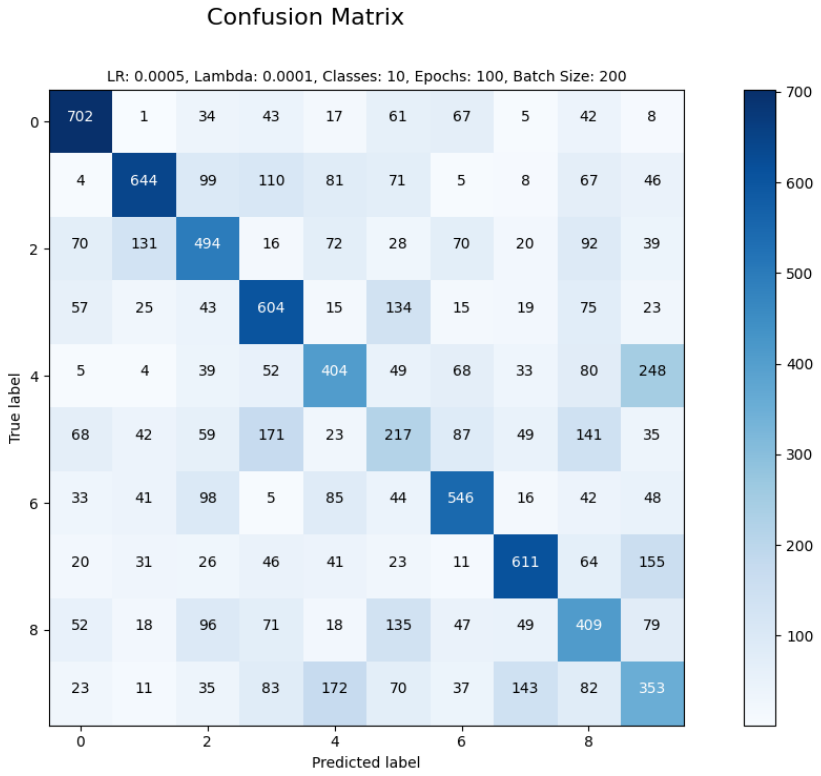
These reconstructions demonstrate PCA's utility in image compression. By keeping only the most significant components, one can approximate the original image with far fewer data points, which is useful in scenarios where storage or transmission bandwidth is limited.

The importance of adding back the mean (which was subtracted during PCA) in the reconstruction process is evident. This step is crucial to ensure the reconstructed image has the correct overall brightness and contrast.

In summary, the number of principal components used in PCA significantly affects the quality of the reconstructed images. While lower-dimensional representations are computationally efficient, they might not capture all the critical features of the original data.

Question 2)

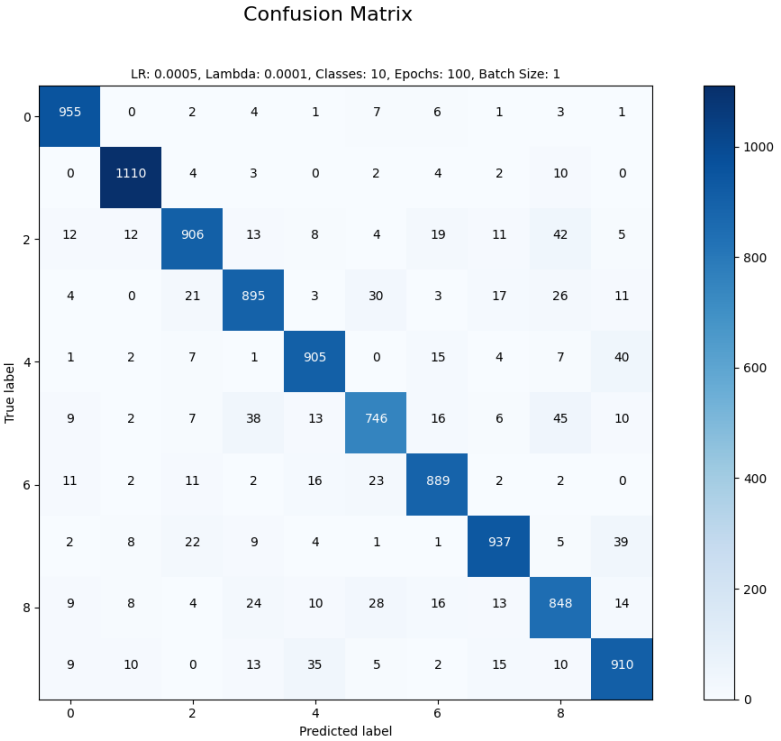
2.1)



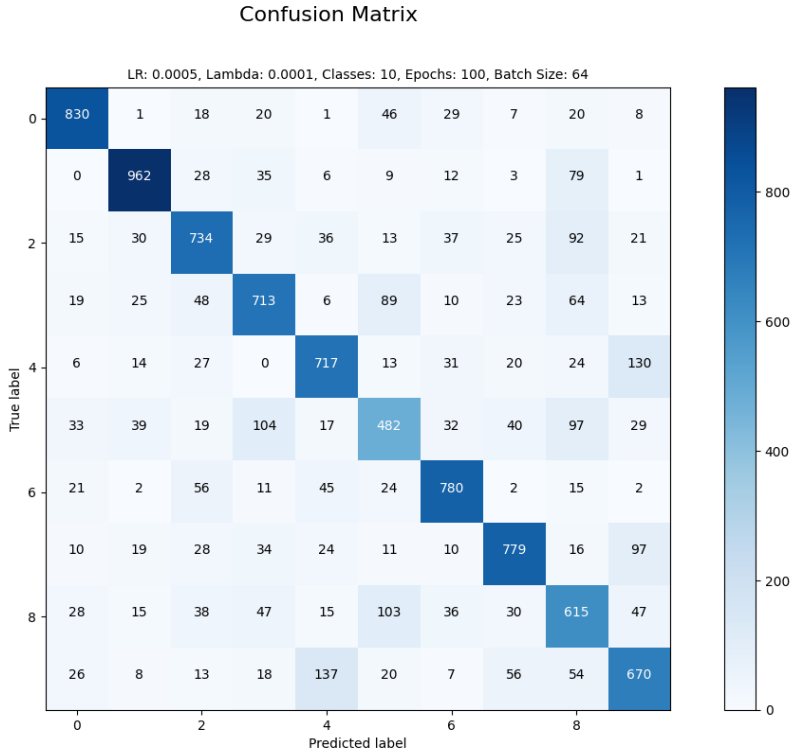
Test Accuracy: 49.84%

2.2)

Batch Size: 1, 64, 50000



BS = 1, test accuracy = 91.01%

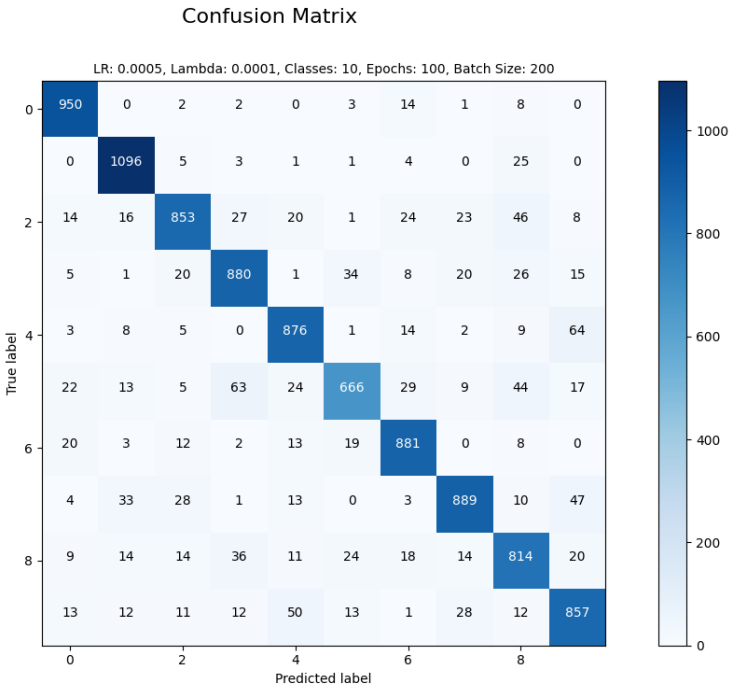


BS = 64, test accuracy = 72.82%

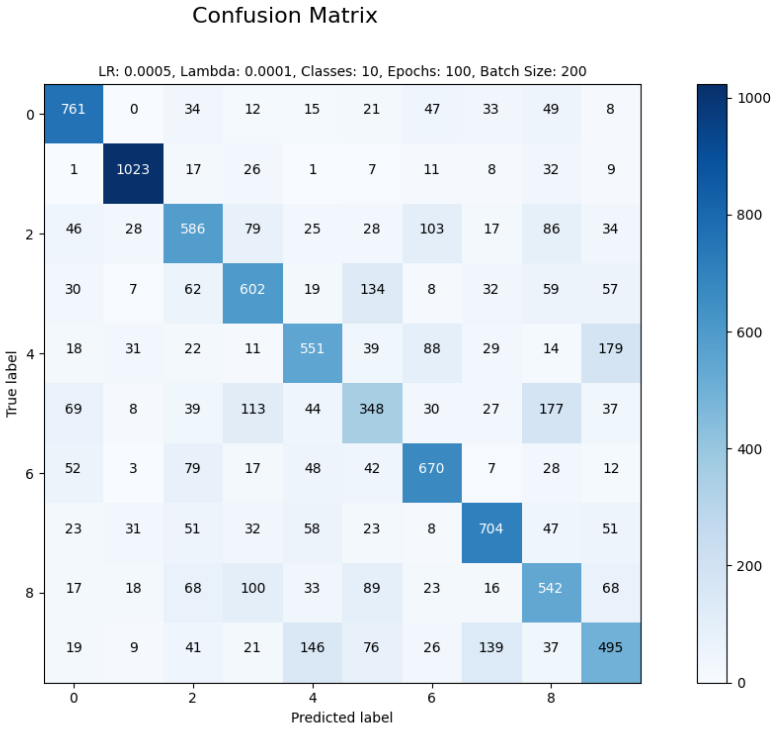
BS = 50000, test accuracy = 8.95%

Comparing the test accuracy performance with different batch sizes, we see that a batch size of 1 achieves the highest accuracy, while larger batch sizes result in lower accuracy. This is likely because smaller batch sizes allow the model to update its weights more frequently, which can help find better generalizing solutions. However, smaller batch sizes can result in longer training times since updates are made after each sample. Larger batch sizes, such as 50000, perform poorly, possibly due to less frequent updates leading to poorer convergence properties. The run time performance generally improves with larger batch sizes since there are fewer updates to perform, but this comes at the cost of accuracy in this case.

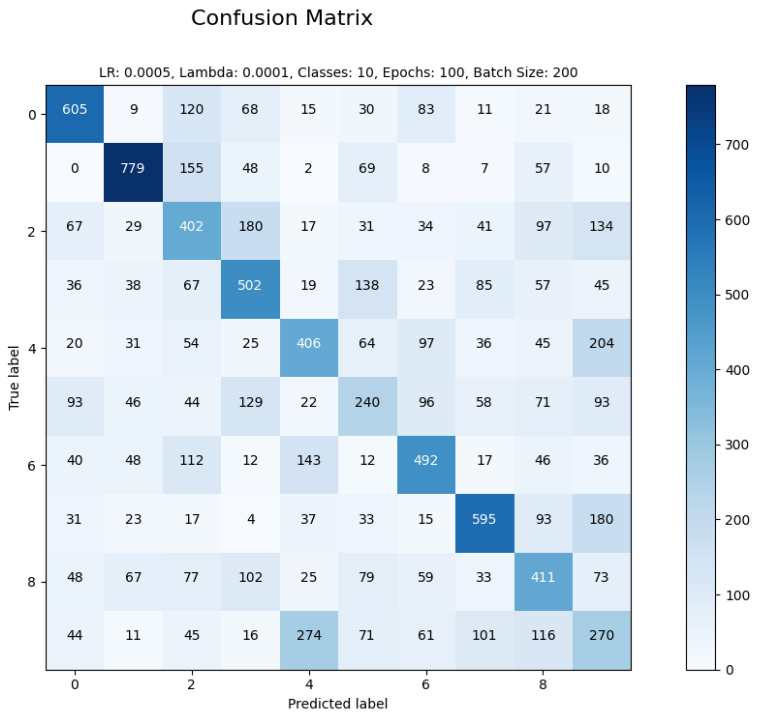
Weight Initialization Technique: Zero Initialization, Uniform Distribution, Normal Distribution



Zero Initialization, test accuracy = 87.62%



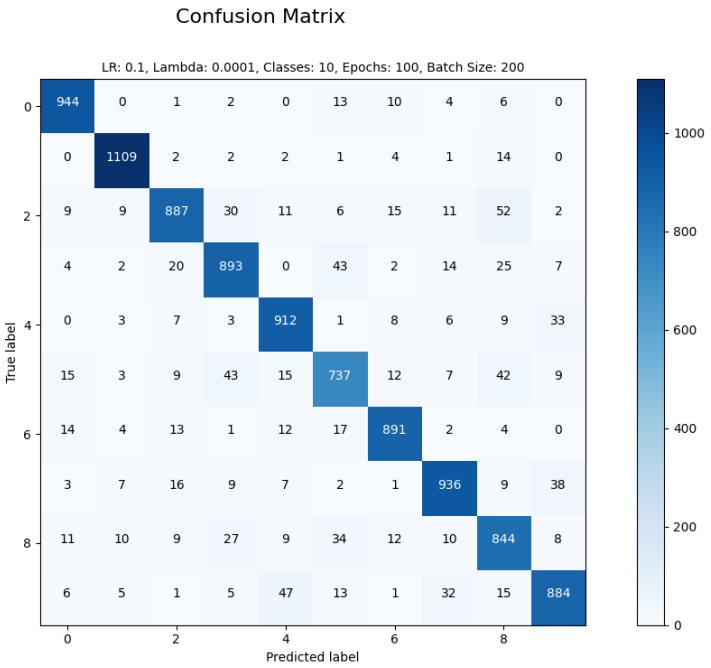
Uniform Distribution, test accuracy = 62.82%



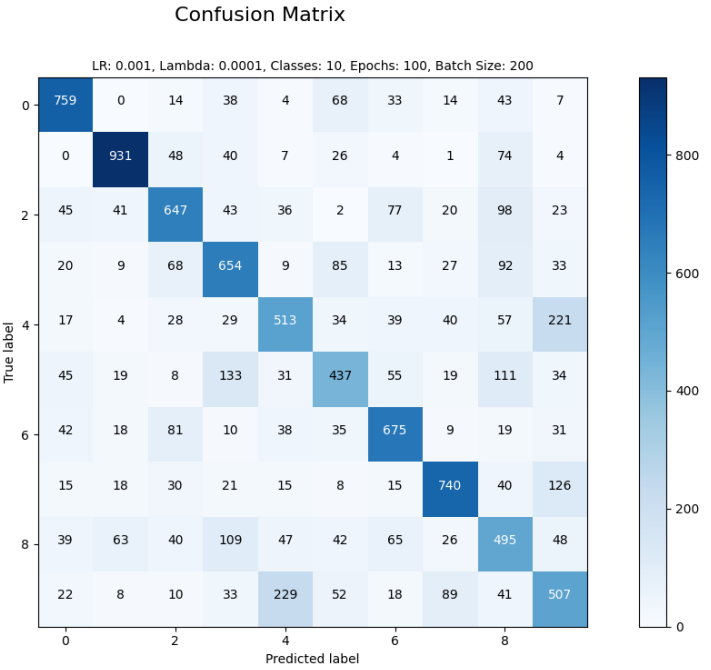
Normal Distribution, test accuracy = 47.02%

Using zero weight initialization leads to the highest accuracy, which is somewhat unusual as zero initialization can sometimes hinder learning by causing symmetry in the gradients. However, in this particular model and dataset, it seems to perform well. Random uniform and normal initializations perform worse, potentially due to poor starting conditions leading to worse local minima. Typically, the runtime is not affected by the weight initialization technique as it is a one-time operation, but the initial weights can significantly impact the speed of convergence and the likelihood of reaching a good local minimum.

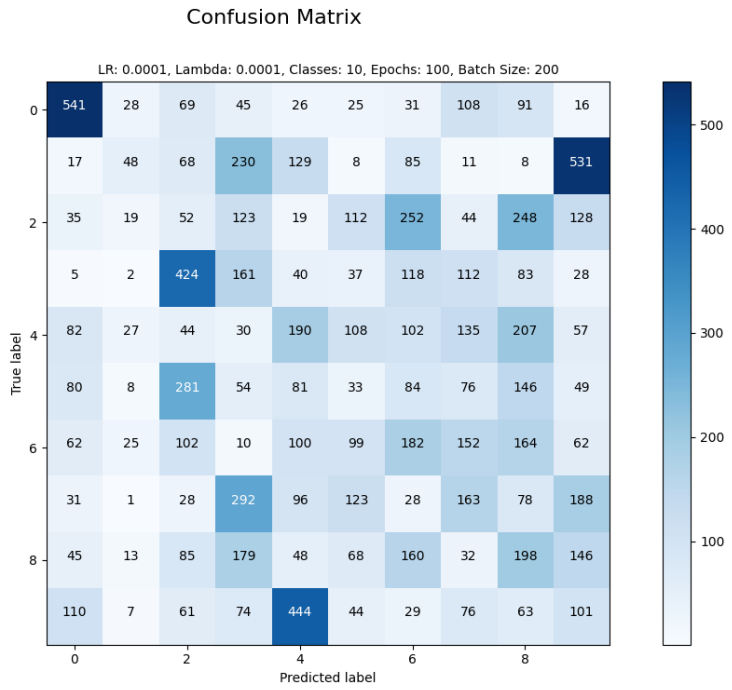
Learning Rate: 0.1, 10^-3, 10^-4, 10^-5



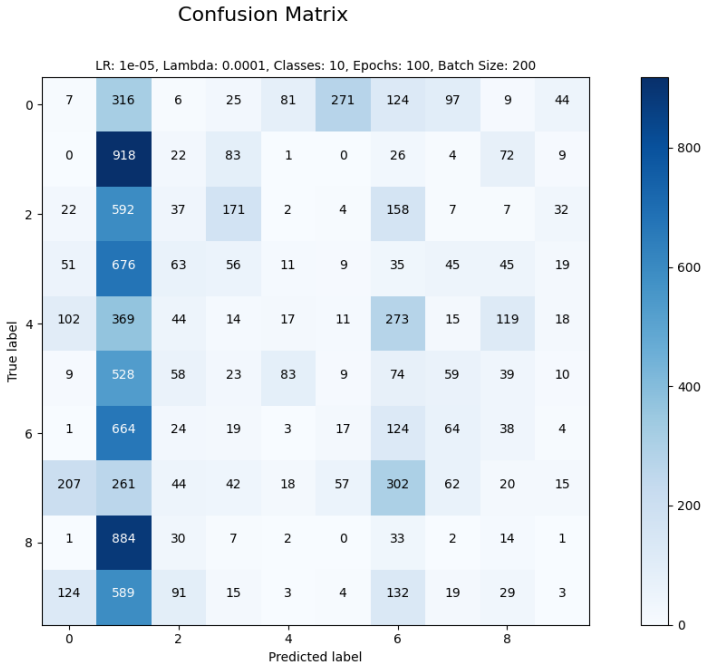
Learning Rate = 0.1, test accuracy = 90.37 %



Learning Rate = 10^-3, test accuracy = 63.58 %



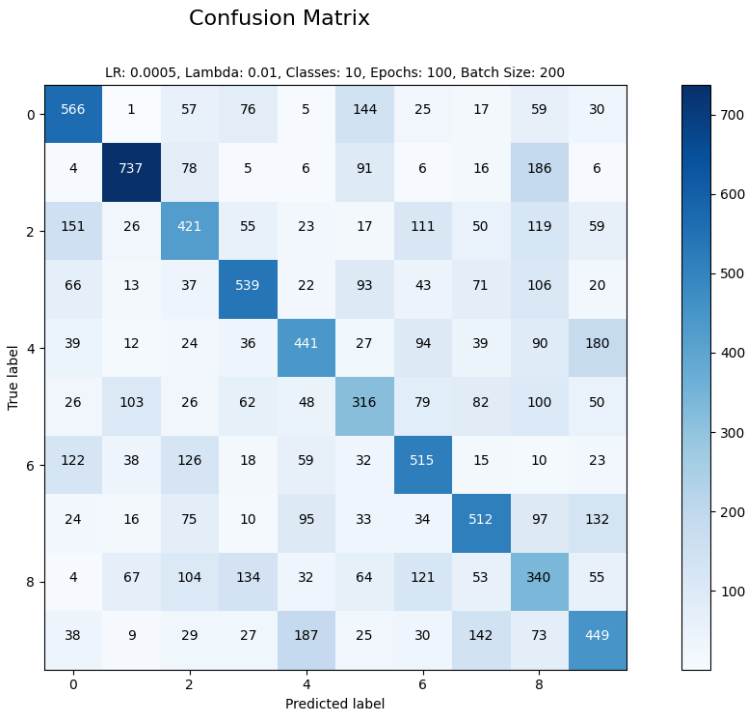
Learning Rate = 10^-4, test accuracy = 18.7 %



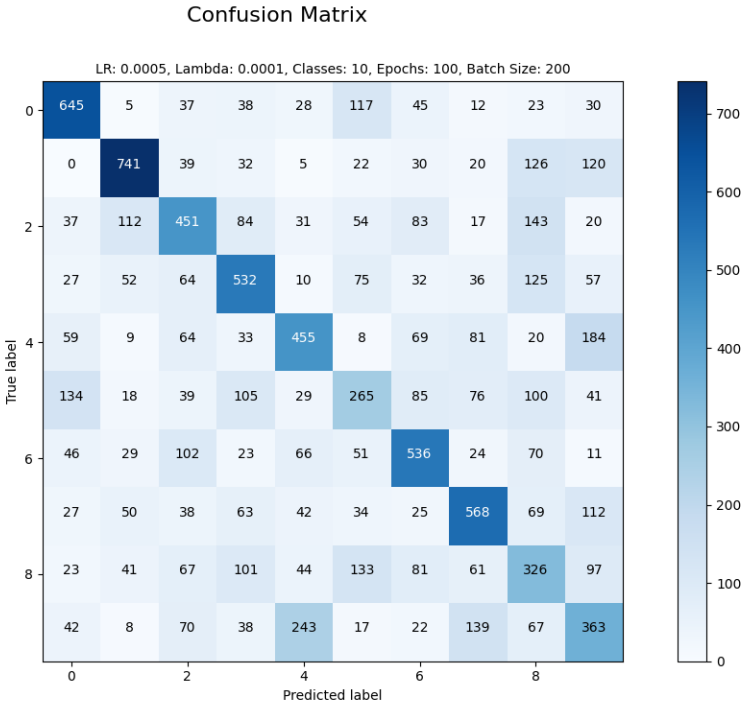
Learning Rate = 10^-5, test accuracy = 12.47 %

The learning rate controls the step size in the weight update process. A learning rate of 0.1 holds the best accuracy. Significantly smaller learning rates, like 10^-5, lead to much lower accuracy, indicating that the model is likely not converging within the given epochs as the updates are too small. Smaller learning rates required more time to train as they make smaller steps towards the minimum of the loss function.

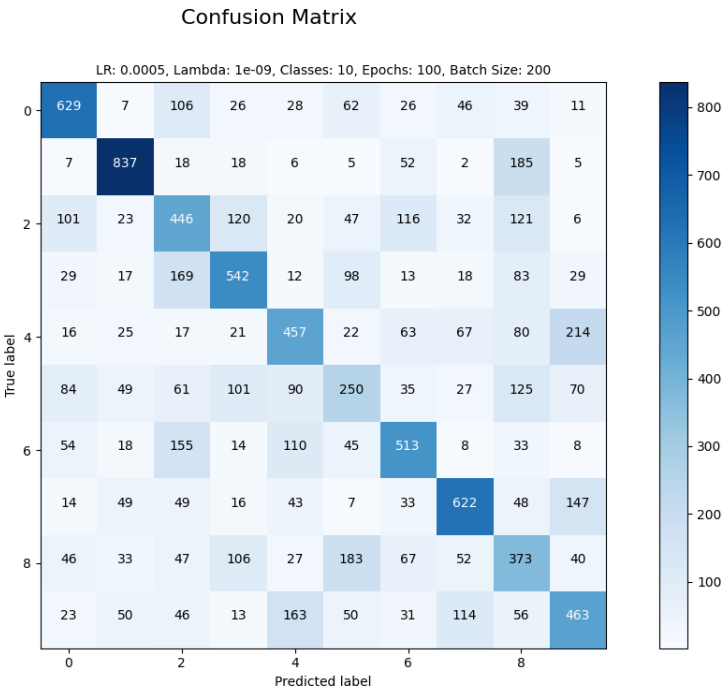
Regulation Coefficient(λ): 10^-2, 10^-4, 10^-9



Regularity Coefficient = 10^-2, test accuracy = 48.36 %



Regularity Coefficient = 10^-4, test accuracy = 48.82 %



Regularity Coefficient = 10^-9, test accuracy = 51.32 %

Regularization helps prevent overfitting by penalizing large weights. The accuracy is fairly close across different regularization coefficients, but the lowest coefficient (10^-9) achieves slightly better performance. This suggests that overfitting is not a major issue with this dataset and model, and a higher penalty on the weights does not necessarily lead to better generalization. A high regularization coefficient can sometimes lead to underfitting, where the model is too biased. The runtime performance is typically not affected by the regularization coefficient as it does not change the complexity of the computation.

2.3)

Best results are taken for each parameter changed once a time are:

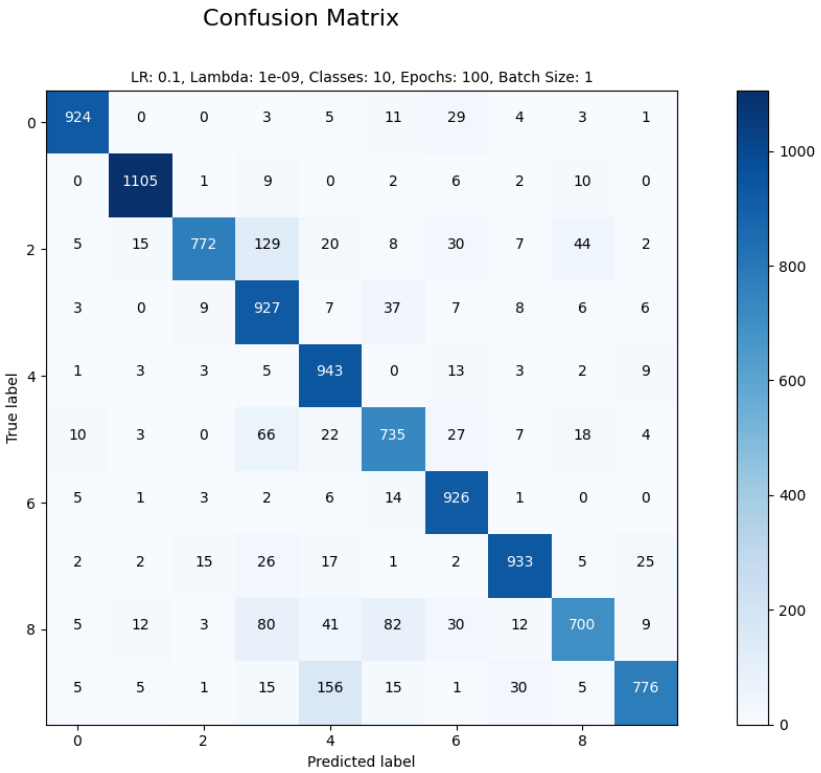
Batch Size = 1, test accuracy = 91.01 %

Learning Rate =0.1, test accuracy = 90.37 %

Regularization coefficient = 10^−9, test accuracy = 51.32%

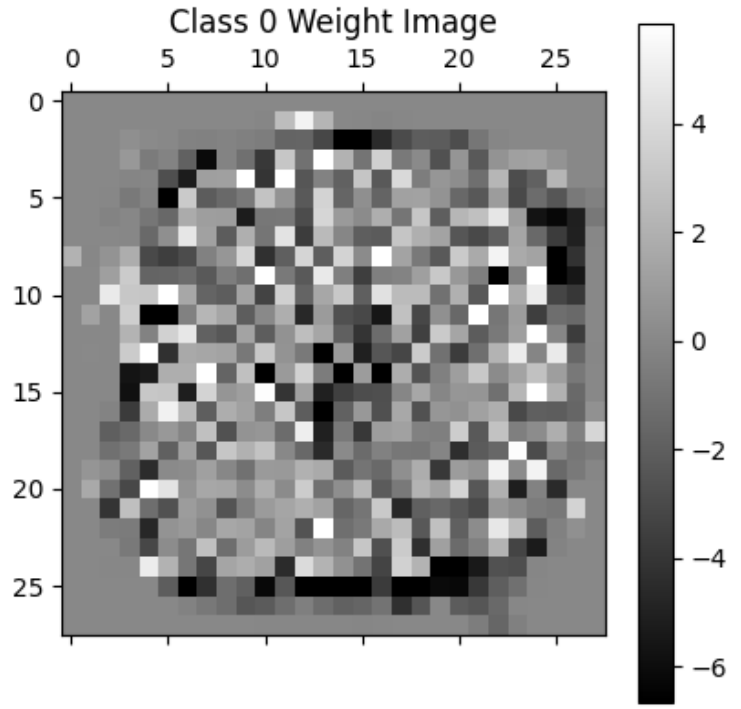
Weight initialization technique = Zero Distribution, test accuracy = 87.62 %

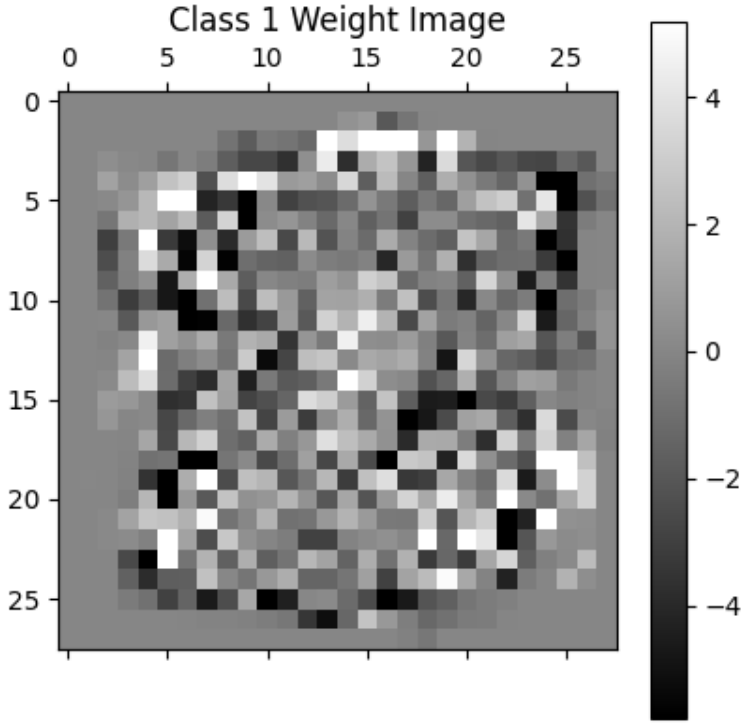
using above parameters we got:

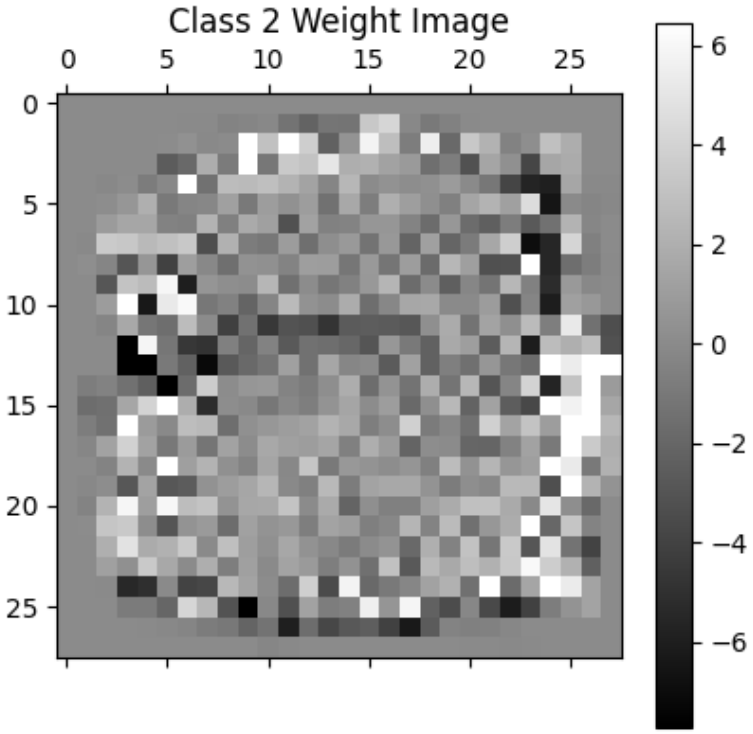


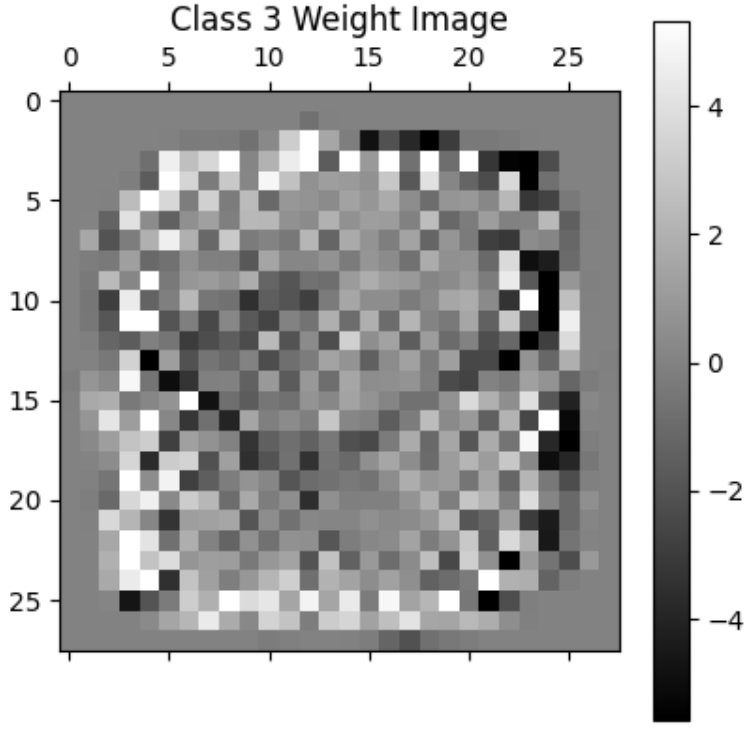
Test accuracy = 89.92 %

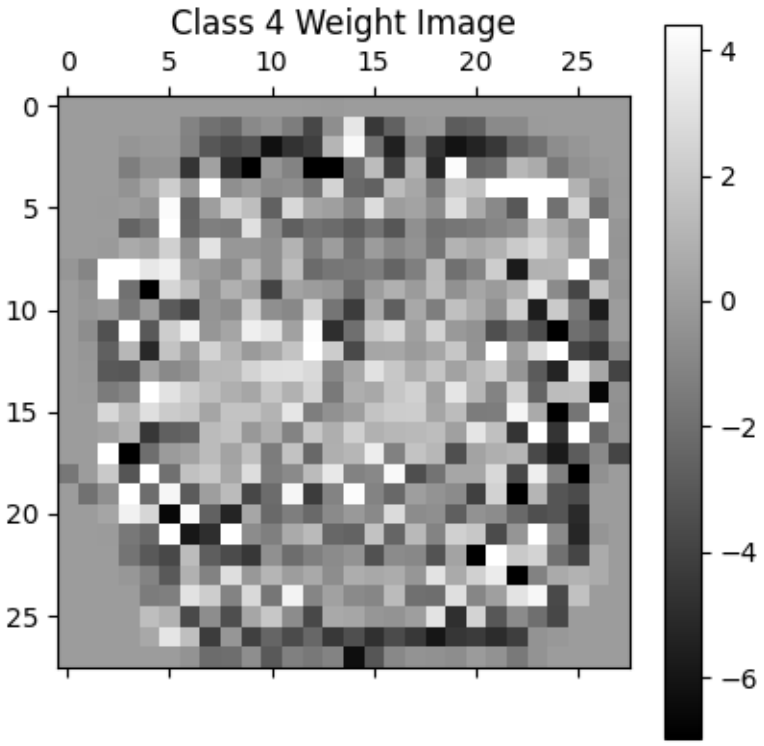
2.4)

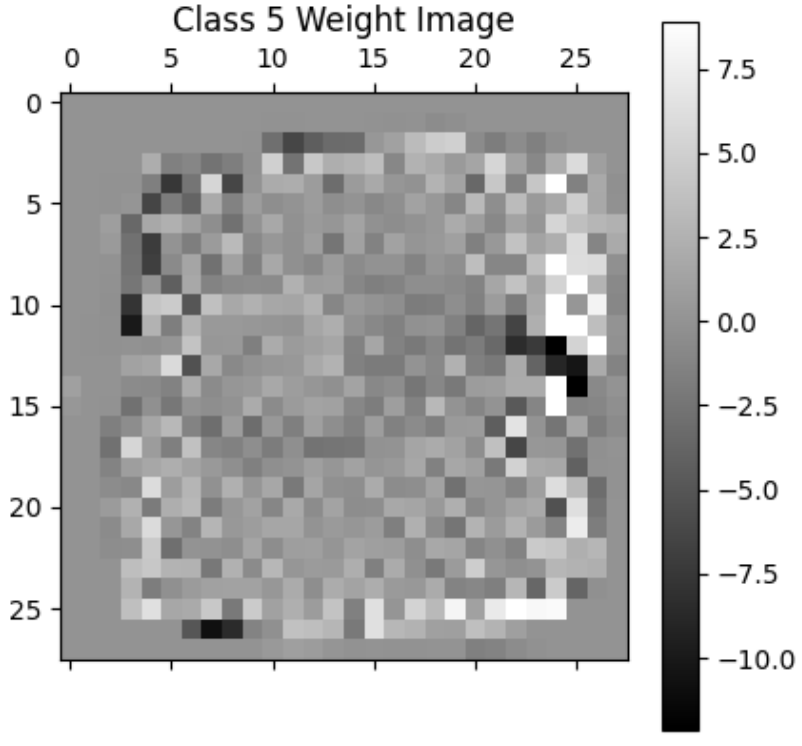

Class 0 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4
−6

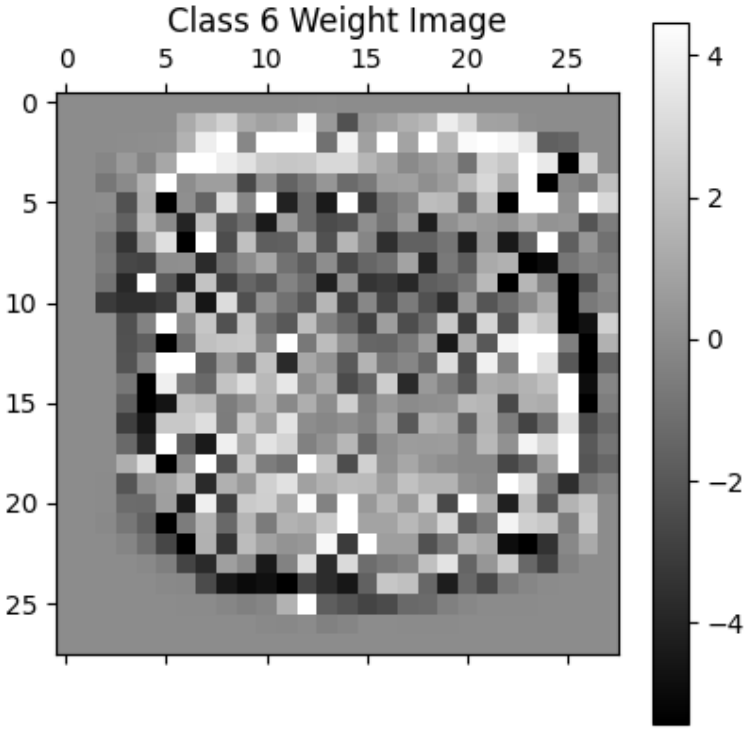

Class 1 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4

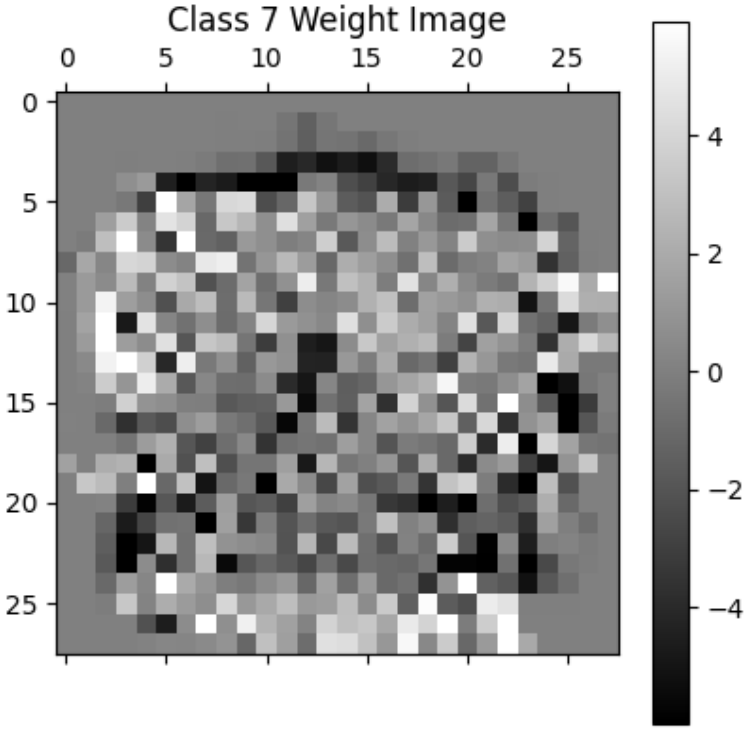

Class 2 Weight Image
0
5
10
15
20
25
6
4
2
0
−2
−4
−6

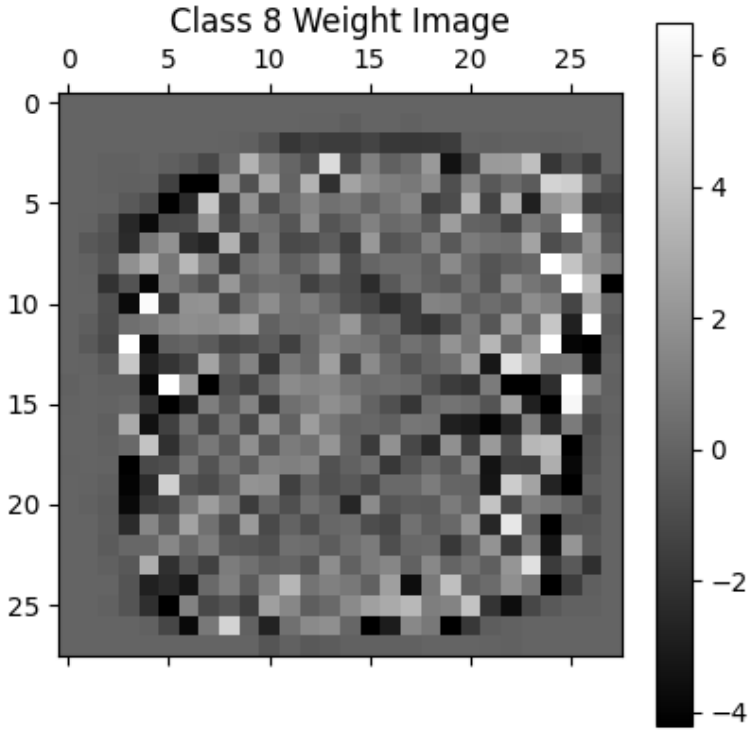

Class 3 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4

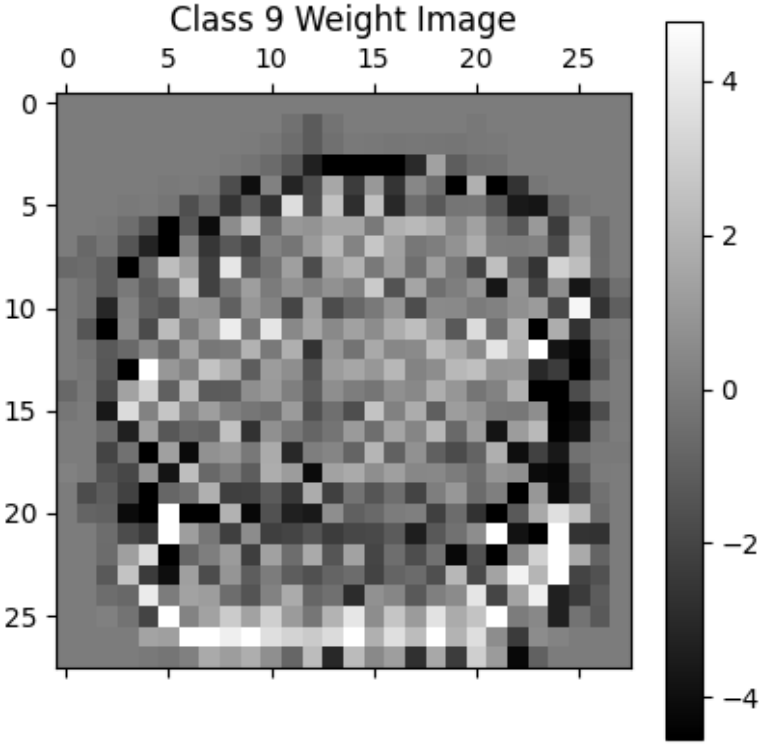

Class 4 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4
−6


Class 5 Weight Image
0
5
10
15
20
25
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0


Class 6 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4


Class 7 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4


Class 8 Weight Image
0
5
10
15
20
25
6
4
2
0
−2
−4


Class 9 Weight Image
0
5
10
15
20
25
4
2
0
−2
−4

The weight images from a machine learning model are visual representations of the model's learned weights for each feature when making predictions about different classes. In this context, each image corresponds to the weights the model uses to identify digits in the MNIST dataset, so they're optimized to represent numerical digits.

- Class 0: This weight image appears to have a clear dark region in the center, which may represent the circular shape of the digit '0'.
- Class 1: This image shows a vertical dark region, which likely corresponds to the straight stroke of the digit '1'.
- Class 2: There seems to be a pattern that could represent the curves at the top and bottom of the digit '2', but it is less clear than for Classes 0 and 1.
- Class 3: The image has several darker areas that could correspond to the rounded parts of the digit '3', although the pattern is not very distinct.
- Class 4: There is a distinct vertical dark line on the right side, which is characteristic of the digit '4', and some horizontal elements that could be the cross stroke.
- Class 5: This weight image is less clear, but there seems to be a horizontal dark line across the top that might represent the top stroke of the digit '5'.
- Class 6: The pattern shows a dark circular area on the bottom right, which might correspond to the loop of the digit '6'.
- Class 7: There is a clear dark horizontal line at the top with a lighter vertical line below it, corresponding to the digit '7'.
- Class 8: This image has multiple dark areas that could represent the two loops of the digit '8', but the pattern is not very distinct.
- Class 9: Similar to Class 6, this image has a dark loop on the top right, which could represent the loop of the digit '9'.

2.5)

We can calculate the values using the following formulas for each class.

$$\text{Precision } P = \frac{TP}{TP+FP}$$

$$\text{Recall } R = \frac{TP}{TP+FN}$$

$$\text{F1 Score } F1 = 2\,\frac{PR}{P+R}$$

F2 Score $F2 = 5\frac{PR}{4P+R}$

The results for each class:

```
Class 0: Precision: 0.97, Recall: 0.94, F1 Score: 0.95, F2 Score: 0.94
Class 1: Precision: 0.97, Recall: 0.98, F1 Score: 0.97, F2 Score: 0.97
Class 2: Precision: 0.95, Recall: 0.84, F1 Score: 0.89, F2 Score: 0.86
Class 3: Precision: 0.89, Recall: 0.89, F1 Score: 0.89, F2 Score: 0.89
Class 4: Precision: 0.90, Recall: 0.92, F1 Score: 0.91, F2 Score: 0.92
Class 5: Precision: 0.92, Recall: 0.72, F1 Score: 0.81, F2 Score: 0.76
Class 6: Precision: 0.86, Recall: 0.96, F1 Score: 0.91, F2 Score: 0.94
Class 7: Precision: 0.92, Recall: 0.92, F1 Score: 0.92, F2 Score: 0.92
Class 8: Precision: 0.78, Recall: 0.88, F1 Score: 0.82, F2 Score: 0.86
Class 9: Precision: 0.85, Recall: 0.92, F1 Score: 0.88, F2 Score: 0.91
```

Precision is high across all classes, particularly for Classes 0 and 1, which suggests that when the model predicts these classes, it is usually correct. This means there are few false positives for these classes.

Recall is lower for some classes, like Class 2 and Class 5, which indicates that the model misses a higher proportion of true positives for these classes. In other words, the model fails to identify all of the actual instances of these classes, leading to false negatives.

F1 Score is the harmonic mean of precision and recall and gives us a single metric that balances the two. Classes 0, 1, 4, and 7 have particularly high F1 scores, which suggests that the model's balance between precision and recall is quite good for these classes.

F2 Score places more emphasis on recall than precision. For Classes 2 and 5, where recall is significantly lower than precision, the F2 score drops, which indicates that the model's tendency to miss true positives is a more severe problem for these classes than incorrectly identifying negatives as positives.

A high number of off-diagonal elements in the rows for Classes 2 and 5 would confirm the lower recall scores, indicating that the model often confuses these classes with others.

Conversely, Classes 0 and 1 likely have few off-diagonals in their rows, indicating that there are few false negatives.