CSC2515 Assignment 3

Hon Wa Ng 1011802284

8 November 2024

Q1(a)

To determine the dimensions of $W^{(1)}$, $W^{(2)}$, z_1, and z_2, I will analyze each part of the computation based on the given information and dimensional requirements:

- **Given Information:**
  - $x \in R^d$ — x is a d-dimensional vector.
  - σ is an activation function applied element-wise.
  - The final output y is a scalar (since the loss $\mathcal{L} = \frac{1}{2}(y-t)^2$ uses $y \in R$).
- **Dimension Analysis for Each Component:**
  - $W^{(1)}$ **and** z_1**:**
    - ∗ Given $z_1 = W^{(1)}x$, $W^{(1)}$ must transform x, a d-dimensional) into another d-dimensional vector.
    - ∗ Therefore, $W^{(1)}$ should be a $d \times d$ matrix (to keep $z_1 \in R^d$) making $z_1 \in R^d$.
  - h**:**
    - ∗ Since $h = \sigma(z_1)$, and $z_1 \in R^d$, applying an activation function element-wise doesn't change the dimensionality.
    - ∗ Thus, $h \in R^d$.
  - z_2**:**
    - ∗ The equation $z_2 = h + x$ is an element-wise addition between h and x, which are both in R^d.
    - ∗ Thus, $z_2 \in R^d$.
  - $W^{(2)}$ **and** y**:**
    - ∗ From the equation $y = W^{(2)}z_2$, and given that y is a scalar (as implied by the loss function), $W^{(2)}$ must map R^d to R.
    - ∗ Therefore, $W^{(2)}$ should be a $1 \times d$ matrix (or row vector) resulting in $y \in R$.

Summary of Dimensions:

- $W^{(1)} \in R^{d\times d}$
- $W^{(2)} \in R^{1\times d}$
- $z_1 \in R^d$
- $z_2 \in R^d$

Q1(b)

I will calculate the number of parameters in this network by determining the number of elements in $W^{(1)}$ and $W^{(2)}$ as a function of d:

- $W^{(1)} \in R^{d\times d}$: This matrix has $d \times d = d^2$ parameters.
- $W^{(2)} \in R^{1\times d}$: This matrix has d parameters (since it is a row vector of length d).

Therefore, the total number of parameters in this network is:

$$d^2 + d$$

Q1(c)

Given the following forward computations:

$$\begin{aligned} z_1 &= W^{(1)}x \\ h &= \sigma(z_1) \\ z_2 &= h + x \\ y &= W^{(2)}z_2 \\ \mathcal{L} &= \frac{1}{2}(y-t)^2 \end{aligned}$$

where:
- σ is the activation function, applied element-wise.
- $t \in R$ is the target.

I aim to compute the gradients of the loss $\mathcal{L}$ with respect to all variables in the network. This includes:

$$\frac{\partial \mathcal{L}}{\partial y}, \quad \frac{\partial \mathcal{L}}{\partial W^{(2)}}, \quad \frac{\partial \mathcal{L}}{\partial z_2}, \quad \frac{\partial \mathcal{L}}{\partial h}, \quad \frac{\partial \mathcal{L}}{\partial z_1}, \quad \frac{\partial \mathcal{L}}{\partial W^{(1)}}, \quad \frac{\partial \mathcal{L}}{\partial x}$$

Gradient with respect to y

The loss function is:

$$\mathcal{L} = \frac{1}{2}(y-t)^2$$

Taking the derivative with respect to y:

$$\frac{\partial \mathcal{L}}{\partial y} = y - t$$

Gradient with respect to $W^{(2)}$

Since $y = W^{(2)} z_2$, by the chain rule:

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial y}{\partial W^{(2)}}$$

From 1, I have $\frac{\partial \mathcal{L}}{\partial y} = y - t$. Since $\frac{\partial y}{\partial W^{(2)}} = z_2^T$, this gives:

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = (y-t) z_2^T$$

Gradient with respect to z_2

Using the chain rule:

$$\frac{\partial \mathcal{L}}{\partial z_2} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial y}{\partial z_2}$$

Since $y = W^{(2)} z_2$, I have $\frac{\partial y}{\partial z_2} = W^{(2)}$. Thus:

$$\frac{\partial \mathcal{L}}{\partial z_2} = (y-t) W^{(2)}$$

Gradient with respect to h

Since $z_2 = h + x$, I have $\frac{\partial z_2}{\partial h} = I$ (the identity matrix). Using the chain rule:

$$\frac{\partial \mathcal{L}}{\partial h} = \frac{\partial \mathcal{L}}{\partial z_2} \cdot \frac{\partial z_2}{\partial h}$$

Substituting $\frac{\partial \mathcal{L}}{\partial z_2} = (y-t) W^{(2)}$, I get:

$$\frac{\partial \mathcal{L}}{\partial h} = (y-t) W^{(2)}$$

Gradient with respect to z_1

Since $h = \sigma(z_1)$, applying the chain rule:

$$\frac{\partial \mathcal{L}}{\partial z_1} = \frac{\partial \mathcal{L}}{\partial h} \cdot \frac{\partial h}{\partial z_1}$$

With $\frac{\partial h}{\partial z_1} = \sigma'(z_1)$ (element-wise derivative of the activation function), I substitute $\frac{\partial \mathcal{L}}{\partial h} = (y-t)W^{(2)}$:

$$\frac{\partial \mathcal{L}}{\partial z_1} = (y-t)W^{(2)}\sigma'(z_1)$$

Gradient with respect to $W^{(1)}$

Given $z_1 = W^{(1)}x$, I use the chain rule:

$$\frac{\partial \mathcal{L}}{\partial W^{(1)}} = \frac{\partial \mathcal{L}}{\partial z_1} \cdot \frac{\partial z_1}{\partial W^{(1)}}$$

Since $\frac{\partial z_1}{\partial W^{(1)}} = x^T$, substituting $\frac{\partial \mathcal{L}}{\partial z_1} = (y-t)W^{(2)}\sigma'(z_1)$, I get:

$$\frac{\partial \mathcal{L}}{\partial W^{(1)}} = (y-t)W^{(2)}\sigma'(z_1)x^T$$

Gradient with respect to x

Since $z_2 = h + x$, the gradient with respect to x has two components (from z_2 and from z_1 through $W^{(1)}$):

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial z_2} + W^{(1)T}\frac{\partial \mathcal{L}}{\partial z_1}$$

Substituting $\frac{\partial \mathcal{L}}{\partial z_2} = (y-t)W^{(2)}$ and $\frac{\partial \mathcal{L}}{\partial z_1} = (y-t)W^{(2)}\sigma'(z_1)$, I get:

$$\frac{\partial \mathcal{L}}{\partial x} = (y-t)W^{(2)} + W^{(1)T}(y-t)W^{(2)}\sigma'(z_1)$$

Summary of Gradients

The final gradients are as follows:

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial y} &= y - t \\
\frac{\partial \mathcal{L}}{\partial W^{(2)}} &= (y - t) z_2^T \\
\frac{\partial \mathcal{L}}{\partial z_2} &= (y - t) W^{(2)} \\
\frac{\partial \mathcal{L}}{\partial h} &= (y - t) W^{(2)} \\
\frac{\partial \mathcal{L}}{\partial z_1} &= (y - t) W^{(2)} \sigma'(z_1) \\
\frac{\partial \mathcal{L}}{\partial W^{(1)}} &= (y - t) W^{(2)} \sigma'(z_1) x^T \\
\frac{\partial \mathcal{L}}{\partial x} &= (y - t) W^{(2)} + W^{(1)T} (y - t) W^{(2)} \sigma'(z_1)
\end{aligned}
$$

Q2(a)

To compute the partial derivative of the softmax output y_k with respect to $z_{k'}$ for any $k, k' = 1, \ldots, K$, I will start with the softmax function given:

$$y_k = \text{softmax}(z)_k = \frac{\exp(z_k)}{\sum_{k'=1}^{K} \exp(z_{k'})}$$

The partial derivative $\frac{\partial y_k}{\partial z_{k'}}$ can be computed as follows:

1. If $k = k'$:

$$\frac{\partial y_k}{\partial z_k} = y_k(1 - y_k)$$

2. If $k \neq k'$:

$$\frac{\partial y_k}{\partial z_{k'}} = -y_k y_{k'}$$

Thus, the result can be written compactly as:

$$\frac{\partial y_k}{\partial z_{k'}} = y_k(\delta_{kk'} - y_{k'})$$

where $\delta_{kk'}$ is the Kronecker delta, equal to 1 if $k = k'$ and 0 otherwise.

Q2(b)

Now, I compute the gradient of the cross-entropy loss $\mathcal{L}_{\text{CE}}$ with respect to the weights $\mathbf{w}_k$.

Given:

$$\mathcal{L}_{\text{CE}}(t, \mathbf{y}) = -\sum_{k=1}^{K} t_k \log y_k$$

Using the chain rule, the partial derivative of $\mathcal{L}_{\text{CE}}$ with respect to $\mathbf{w}_k$ is:

$$\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \mathbf{w}_k} = \frac{\partial \mathcal{L}_{\text{CE}}}{\partial y_k} \cdot \frac{\partial y_k}{\partial z_k} \cdot \frac{\partial z_k}{\partial \mathbf{w}_k}$$

Breaking down each term:
1. **Loss derivative with respect to y_k:**

$$\frac{\partial \mathcal{L}_{\text{CE}}}{\partial y_k} = -\frac{t_k}{y_k}$$

2. **Softmax derivative (from part (a)):**

$$\frac{\partial y_k}{\partial z_k} = y_k(1 - y_k) \quad \text{and for } k \neq k', \quad \frac{\partial y_k}{\partial z_{k'}} = -y_k y_{k'}$$

3. **Derivative of z_k with respect to $\mathbf{w}_k$:** Since $z_k = \mathbf{w}_k^\top \mathbf{x}$, I have:

$$\frac{\partial z_k}{\partial \mathbf{w}_k} = \mathbf{x}$$

Combining these:

$$\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \mathbf{w}_k} = (y_k - t_k)\,\mathbf{x}$$

This result provides the gradient update for each row of $\mathbf{W}$ in softmax regression.

Q3.1

To derive an expression for $p(y = k|\mathbf{x}, \mu, \sigma)$ using Bayes' rule, I start with:

$$p(y = k|\mathbf{x}, \mu, \sigma) = \frac{p(\mathbf{x}|y = k, \mu, \sigma)p(y = k)}{p(\mathbf{x}|\mu, \sigma)}$$

Likelihood

The likelihood for $p(\mathbf{x}|y = k, \mu, \sigma)$ is given by:

$$p(\mathbf{x}|y = k, \mu, \sigma) = \prod_{i=1}^{D} \left(\frac{1}{\sqrt{2\pi\sigma_i^2}} \right) \exp\left(-\frac{1}{2} \sum_{i=1}^{D} \frac{(x_i - \mu_{ki})^2}{\sigma_i^2} \right)$$

Prior

The prior for class k is:

$$p(y = k) = \alpha_k$$

Evidence

The evidence is:

$$p(\mathbf{x}|\mu,\sigma) = \sum_{j=1}^{K} p(\mathbf{x}|y=j,\mu,\sigma)p(y=j)$$

Final Expression

Substituting these into Bayes' rule:

$$p(y=k|\mathbf{x},\mu,\sigma) = \frac{\alpha_k \exp\left(-\sum_{i=1}^{D}\frac{(x_i-\mu_{ki})^2}{2\sigma_i^2}\right)}{\sum_{j=1}^{K}\alpha_j \exp\left(-\sum_{i=1}^{D}\frac{(x_i-\mu_{ji})^2}{2\sigma_i^2}\right)}$$

Q3.2

The negative log-likelihood (NLL) function for a dataset $D = \{(y^{(n)}, \mathbf{x}^{(n)})\}_{n=1}^{N}$, given the parameters $\theta = \{\alpha, \mu, \sigma\}$, is defined as:

$$\mathcal{L}(\theta; D) = -\log p(y^{(1)}, \mathbf{x}^{(1)}, y^{(2)}, \mathbf{x}^{(2)}, \ldots, y^{(N)}, \mathbf{x}^{(N)}|\theta)$$

Assuming that the data points are independent and identically distributed (i.i.d.), this can be rewritten as:

$$\mathcal{L}(\theta; D) = -\sum_{n=1}^{N} \log p(y^{(n)}, \mathbf{x}^{(n)}|\theta)$$

Expanding $p(y^{(n)}, \mathbf{x}^{(n)}|\theta)$ using the chain rule:

$$p(y^{(n)}, \mathbf{x}^{(n)}|\theta) = p(\mathbf{x}^{(n)}|y^{(n)}, \mu, \sigma)p(y^{(n)}|\alpha)$$

Thus:

$$\mathcal{L}(\theta; D) = -\sum_{n=1}^{N}\left(\log p(\mathbf{x}^{(n)}|y^{(n)}, \mu, \sigma) + \log \alpha_{y^{(n)}}\right)$$

Substituting the Gaussian likelihood for $p(\mathbf{x}^{(n)}|y^{(n)}, \mu, \sigma)$ from the previous step:

$$\mathcal{L}(\theta; D) = -\sum_{n=1}^{N}\left(\sum_{i=1}^{D}\left(-\frac{1}{2}\log(2\pi\sigma_i^2) - \frac{(x_i^{(n)} - \mu_{y^{(n)}i})^2}{2\sigma_i^2}\right) + \log \alpha_{y^{(n)}}\right)$$

Simplifying, the NLL becomes:

$$\mathcal{L}(\theta; D) = \frac{ND}{2}\log(2\pi) + \frac{N}{2}\sum_{i=1}^{D}\log\sigma_i^2 + \sum_{n=1}^{N}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{y^{(n)}i})^2}{2\sigma_i^2} - \sum_{n=1}^{N}\log\alpha_{y^{(n)}}$$

This expression represents the negative log-likelihood function for the given dataset and parameters.

Q3.3

To compute the partial derivatives of the negative log-likelihood $\mathcal{L}(\theta; D)$ with respect to the parameters μ_{ki} and σ_i^2, I start with the simplified form of the NLL function derived in question 3b:

$$\mathcal{L}(\theta; D) = \frac{ND}{2}\log(2\pi) + \frac{N}{2}\sum_{i=1}^{D}\log\sigma_i^2 + \sum_{n=1}^{N}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{y^{(n)}i})^2}{2\sigma_i^2} - \sum_{n=1}^{N}\log\alpha_{y^{(n)}}$$

The parameters of interest are μ_{ki} (the mean of feature i for class k) and σ_i^2 (the shared variance for feature i across all classes).

Partial Derivative with Respect to μ_{ki}

To find the partial derivative of $\mathcal{L}(\theta; D)$ with respect to μ_{ki}, the third term in the NLL expression is focused - as it is the only term that depends on μ_{ki}:

$$\sum_{n=1}^{N}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{y^{(n)}i})^2}{2\sigma_i^2}$$

Expanding this term to isolate the dependence on μ_{ki}:

$$\sum_{n=1}^{N}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{y^{(n)}i})^2}{2\sigma_i^2} = \sum_{k=1}^{K}\sum_{n\in C_k}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{ki})^2}{2\sigma_i^2}$$

where C_k denotes the set of indices n for which $y^{(n)} = k$.
Now, to take the partial derivative with respect to μ_{ki}:

$$\frac{\partial\mathcal{L}(\theta; D)}{\partial\mu_{ki}} = \frac{\partial}{\partial\mu_{ki}}\sum_{k=1}^{K}\sum_{n\in C_k}\sum_{i=1}^{D}\frac{(x_i^{(n)} - \mu_{ki})^2}{2\sigma_i^2}$$

Focusing on the relevant terms:

$$\frac{\partial\mathcal{L}(\theta; D)}{\partial\mu_{ki}} = \sum_{n\in C_k}\frac{-(x_i^{(n)} - \mu_{ki})}{\sigma_i^2}$$

Simplifying further:

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \mu_{ki}} = -\frac{1}{\sigma_i^2} \sum_{n \in C_k} (x_i^{(n)} - \mu_{ki})$$

Partial Derivative with Respect to σ_i^2

Next, I compute the partial derivative of $\mathcal{L}(\theta; D)$ with respect to σ_i^2. Here, both the second and third terms of the NLL depend on σ_i^2:

$$\frac{N}{2} \sum_{i=1}^{D} \log \sigma_i^2 + \sum_{n=1}^{N} \sum_{i=1}^{D} \frac{(x_i^{(n)} - \mu_{y^{(n)} i})^2}{2\sigma_i^2}$$

Taking the partial derivative with respect to σ_i^2:

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \sigma_i^2} = \frac{N}{2\sigma_i^2} - \sum_{n=1}^{N} \frac{(x_i^{(n)} - \mu_{y^{(n)} i})^2}{2(\sigma_i^2)^2}$$

Rearranging terms:

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \sigma_i^2} = \frac{N}{2\sigma_i^2} - \frac{1}{2(\sigma_i^2)^2} \sum_{n=1}^{N} (x_i^{(n)} - \mu_{y^{(n)} i})^2$$

Summary of Results

The partial derivatives of the negative log-likelihood $\mathcal{L}(\theta; D)$ with respect to μ_{ki} and σ_i^2 are:

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \mu_{ki}} = -\frac{1}{\sigma_i^2} \sum_{n \in C_k} (x_i^{(n)} - \mu_{ki})$$

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \sigma_i^2} = \frac{N}{2\sigma_i^2} - \frac{1}{2(\sigma_i^2)^2} \sum_{n=1}^{N} (x_i^{(n)} - \mu_{y^{(n)} i})^2$$

These derivatives will be used to find the maximum likelihood estimates for μ and σ in question 3d.

Q3.4

To find the maximum likelihood estimates (MLE) for μ_{ki} and σ_i^2, I set the partial derivatives of the negative log-likelihood (NLL) function $\mathcal{L}(\theta; D)$ with respect to each parameter equal to zero.

From question 3.3:

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \mu_{ki}} = -\frac{1}{\sigma_i^2} \sum_{n \in C_k} (x_i^{(n)} - \mu_{ki}) = 0$$

$$\frac{\partial \mathcal{L}(\theta; D)}{\partial \sigma_i^2} = \frac{N}{2\sigma_i^2} - \frac{1}{2(\sigma_i^2)^2}\sum_{n=1}^{N}(x_i^{(n)} - \mu_{y^{(n)}i})^2 = 0$$

where C_k denotes the set of indices n for which $y^{(n)} = k$.
Maximum Likelihood Estimate for μ_{ki}
Setting the partial derivative with respect to μ_{ki} to zero:

$$-\frac{1}{\sigma_i^2}\sum_{n \in C_k}(x_i^{(n)} - \mu_{ki}) = 0$$

Solving for μ_{ki}:

$$\sum_{n \in C_k}(x_i^{(n)} - \mu_{ki}) = 0$$

$$\mu_{ki} = \frac{\sum_{n \in C_k} x_i^{(n)}}{|C_k|}$$

where $|C_k|$ is the number of samples in class k. Thus, the MLE for μ_{ki} is the sample mean of feature i for all samples in class k:

$$\hat{\mu}_{ki} = \frac{\sum_{n \in C_k} x_i^{(n)}}{|C_k|}$$

Maximum Likelihood Estimate for σ_i^2
Now, to the partial derivative with respect to σ_i^2 to zero:

$$\frac{N}{2\sigma_i^2} - \frac{1}{2(\sigma_i^2)^2}\sum_{n=1}^{N}(x_i^{(n)} - \mu_{y^{(n)}i})^2 = 0$$

Multiplying both sides by $2(\sigma_i^2)^2$ to clear the denominator:

$$N\sigma_i^2 = \sum_{n=1}^{N}(x_i^{(n)} - \mu_{y^{(n)}i})^2$$

Solving for σ_i^2:

$$\sigma_i^2 = \frac{1}{N}\sum_{n=1}^{N}(x_i^{(n)} - \mu_{y^{(n)}i})^2$$

Thus, the MLE for σ_i^2 is the average of the squared deviations of feature i across all classes.

To summarize, the maximum likelihood estimates for μ_{ki} and σ_i^2 are:

$$\hat{\mu}_{ki} = \frac{\sum_{n \in C_k} x_i^{(n)}}{|C_k|}$$

$$\hat{\sigma}_i^2 = \frac{1}{N}\sum_{n=1}^{N}(x_i^{(n)} - \mu_{y^{(n)}i})^2$$

where: (1) $\hat{\mu}_{ki}$ is the sample mean of feature i for class k. and (2)$\hat{\sigma}_i^2$ is the shared variance for feature i across all classes.

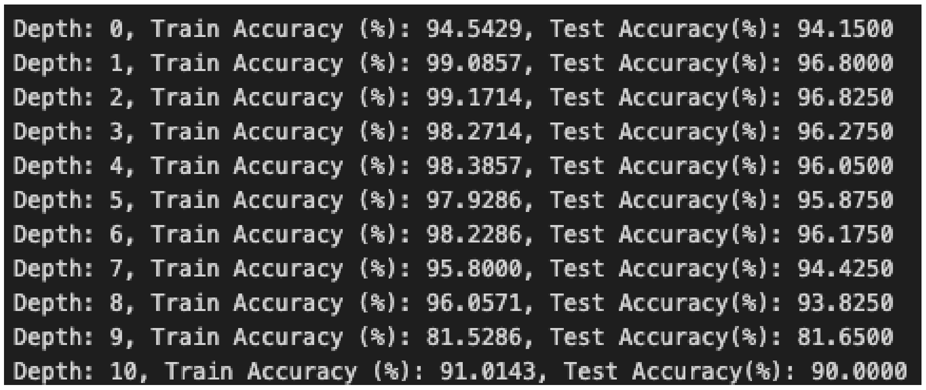
These are the maximum likelihood estimates for μ and σ.

Q4.1a

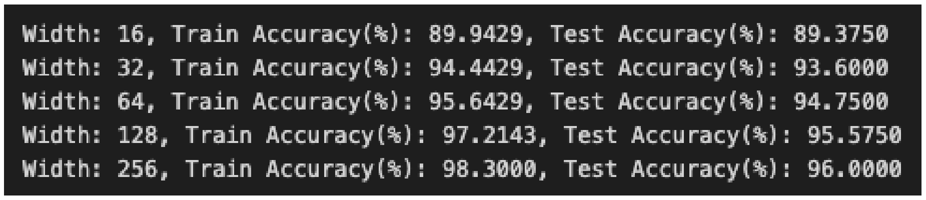
Mean Images for Each Digit Class (0-9)

The figure shows the average or mean image for each digit class (0 to 9) calculated from the training data.

Q4.1b



```
Depth: 0, Train Accuracy (%): 94.5429, Test Accuracy(%): 94.1500
Depth: 1, Train Accuracy (%): 99.0857, Test Accuracy(%): 96.8000
Depth: 2, Train Accuracy (%): 99.1714, Test Accuracy(%): 96.8250
Depth: 3, Train Accuracy (%): 98.2714, Test Accuracy(%): 96.2750
Depth: 4, Train Accuracy (%): 98.3857, Test Accuracy(%): 96.0500
Depth: 5, Train Accuracy (%): 97.9286, Test Accuracy(%): 95.8750
Depth: 6, Train Accuracy (%): 98.2286, Test Accuracy(%): 96.1750
Depth: 7, Train Accuracy (%): 95.8000, Test Accuracy(%): 94.4250
Depth: 8, Train Accuracy (%): 96.0571, Test Accuracy(%): 93.8250
Depth: 9, Train Accuracy (%): 81.5286, Test Accuracy(%): 81.6500
Depth: 10, Train Accuracy (%): 91.0143, Test Accuracy(%): 90.0000
```



```
Width: 16, Train Accuracy(%): 89.9429, Test Accuracy(%): 89.3750
Width: 32, Train Accuracy(%): 94.4429, Test Accuracy(%): 93.6000
Width: 64, Train Accuracy(%): 95.6429, Test Accuracy(%): 94.7500
Width: 128, Train Accuracy(%): 97.2143, Test Accuracy(%): 95.5750
Width: 256, Train Accuracy(%): 98.3000, Test Accuracy(%): 96.0000
```

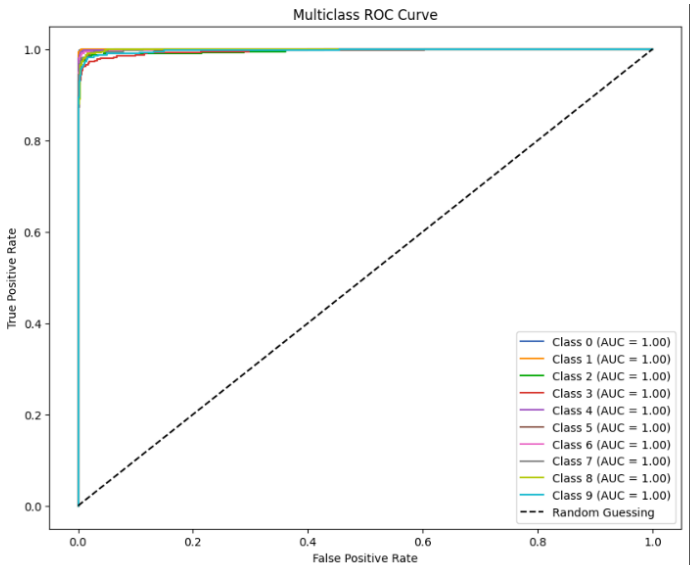
Q4.1c

Optimal Dropout Rate is 0.1 with Validation Accuracy: 94.44

Q4.1d

1. **ROC (Receiver Operating Characteristics) Curve**

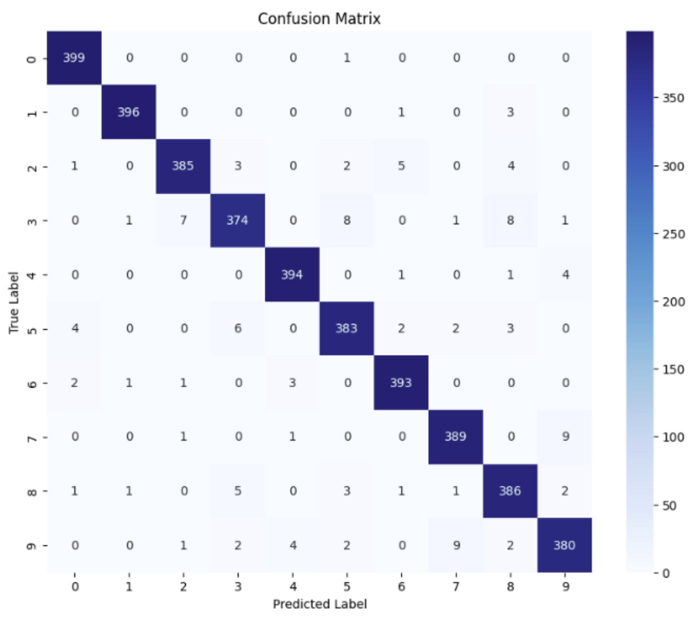
    - **Description**: The ROC curve shows the relationship between the True Positive Rate (sensitivity) and False Positive Rate for different threshold settings. The area under the ROC curve (AUC) provides a single measure of the model's ability to distinguish between classes, with an AUC of 1 indicating perfect classification.
    - **Reported Value**: The model achieved an AUC of 1.00 for each class, indicating perfect separation between the classes.



2. **Confusion Matrix**

    - **Description**: The confusion matrix provides a summary of prediction results on a classification problem, with rows representing the actual classes and columns representing the predicted classes. It shows the counts of true positives, false positives, false negatives, and true negatives for each class.
    - **Reported Value**: The confusion matrix reveals that most predictions fall along the diagonal (indicating correct classifications), with a few off-diagonal entries (misclassifications). The model has a high degree of accuracy, as shown by the strong diagonal pattern.

3. **Accuracy**



- **Description**: Accuracy is the ratio of correctly predicted instances to the total instances and provides an overall measure of the model's performance.
- **Reported Value**: The model's accuracy is 96.98%, which indicates that approximately 97% of the predictions are correct.

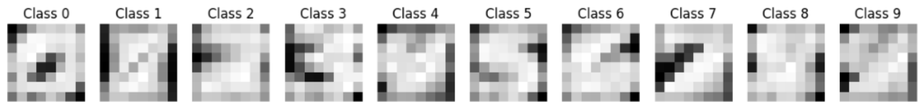
4. **Precision**

   - **Description**: Precision, calculated as Precision $= \frac{\text{True Positives}}{\text{True Positives}+\text{False Positives}}$, measures the proportion of positive identifications that were actually correct. It is particularly important when the cost of false positives is high.
   - **Reported Value**: The model's precision is 96.97%, suggesting that nearly 97% of the model's positive predictions were correct.

5. **Recall**

   - **Description**: Recall, calculated as Recall $= \frac{\text{True Positives}}{\text{True Positives}+\text{False Negatives}}$, measures the proportion of actual positives that were correctly identified. It is important in scenarios where missing a positive case has high consequences.
   - **Reported Value**: The model's recall is 96.98%, indicating that it successfully identified approximately 97% of the actual positive cases.

Q4.2a



Q4.2c

Average Conditional Log-Likelihood (Train): -0.12458797208729135
Average Conditional Log-Likelihood (Test): -0.19660908967370427

Q4.2d

Classification Accuracy (Train): 98.14%
Classification Accuracy (Test): 97.28%

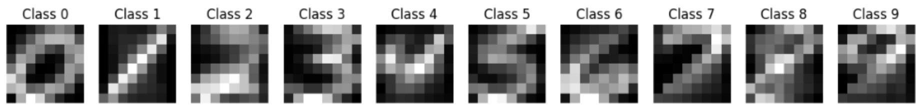
Q4.3a

Please refer to the code.

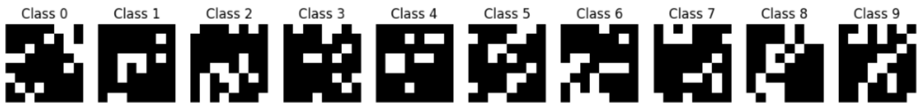
Q4.3b

Please refer to the code.

Q4.3c



Q4.3d



Q4.3e

Average conditional log-likelihood (train): -0.9437538618002556
Average conditional log-likelihood (test): -0.9872704337253582

Q4.3f

Train accuracy: 77.41%
Test accuracy: 76.42%