Assignment 03: Convolutional Neural Networks (CNNs) Solution Report

Tazmeen Afroz
Student ID: 22P-9252
Course: ANN

1 Task 0: Understanding the Working Example

1.1 Question 1: Array Sizes

Question: What are the sizes of the arrays feature_map, activated, pooled, flattened, dense_output, and output_layer?

Solution:

- feature_map: (3, 3)
- activated: (3, 3)
- pooled: (2, 2)
- flattened: (4,)
- dense_output: (10,)
- output_layer: (10,)

1.2 Question 2: Expected Values in 1st Epoch

Question: Provide the expected values of arrays feature_map and output_layer in the 1st epoch.

Solution:

- feature_map values: [[1.04270559 3.40476972 0.89173121] [0.39501705 3.54303402 0.39501705] [-0.07445733 4.31046875 1.97422987]]
- output_layer values: [0.10207149 0.1029142 0.11063334 0.1040532 0.09178496 0.09908408 0.09643855 0.09856632 0.09880543 0.09564843]

1.3 Question 3: Expected Values in Last Epoch

Question: Provide the expected values of arrays feature_map and output_layer in the last epoch.

Solution:

- feature_map values: [[1.37457737 4.66538004 0.11070414] [-1.3449602 4.92962941 -1.3449602] [-0.6315913 4.70716214 -0.86014491]]
- output_layer values: [0.11555551 0.07818263 0.08202687 0.1039702 0.09318277 0.11529241 0.11676104 0.08458151 0.1073446 0.10310246]

1.4 Question 4: Addition of 1e-9 in Loss Function

Question: Explain why I have added 1e-9 to the output_layer when calculating the loss function?

Solution: The $\log(\hat{y}_i)$ term in the cross-entropy loss computation can cause numerical instability if any element $\hat{y}_i$ in the output layer is exactly 0, since $\log(0)$ is undefined

and approaches $-\infty$. To prevent this, a small constant (10^{-9}) is added to $\hat{y}_i$, ensuring no element is zero. For example, $\log(10^{-9})$ is a large negative number but finite, thus maintaining numerical computations without overflow.

1.5 Question 5: Learning Rate Comparison

Question: Present a single plot showing the avg_loss against epochs for learning rate = 0.1, 0.01, 0.001, and 0.0001.

Solution:

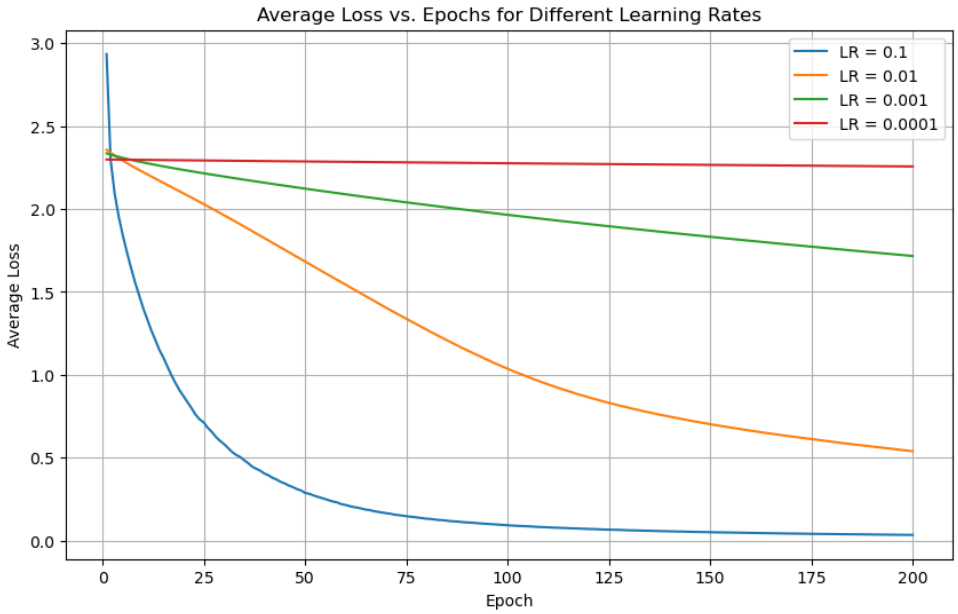


Figure 1: Average loss vs epochs for different learning rates

1.6 Question 6: Kernel Size Comparison

Question: What is the loss at 200th epoch for kernel sizes of 2×2, 3×3, and 5×5?

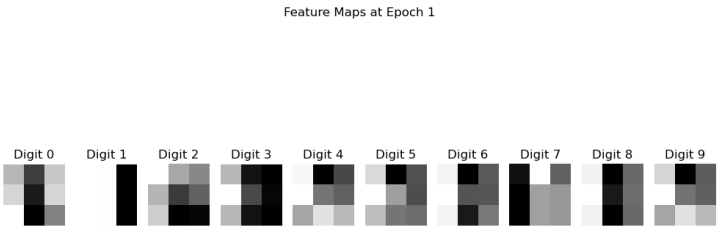
Solution:

Kernel Size	Loss at 200th Epoch
2×2	0.1743
3×3	0.0122
5×5	1.5768

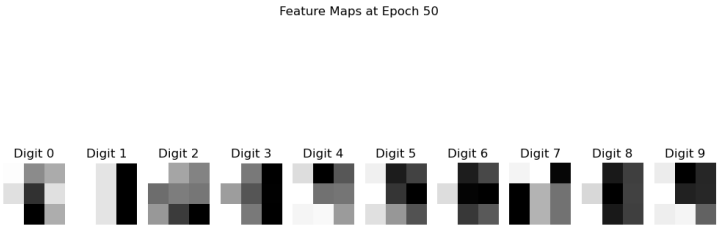
Table 1: Loss comparison for different kernel sizes

1.7 Question 7: Feature Map Visualization

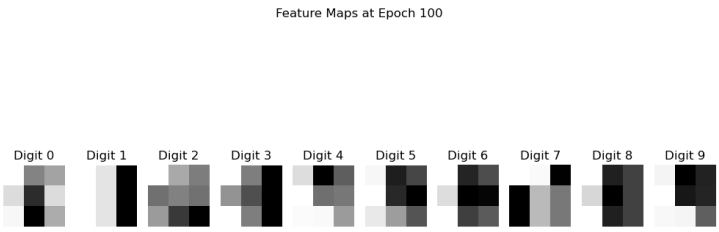
Question: Inspect feature_map at epochs 1, 50, 100, 150, and 200. **Solution:**



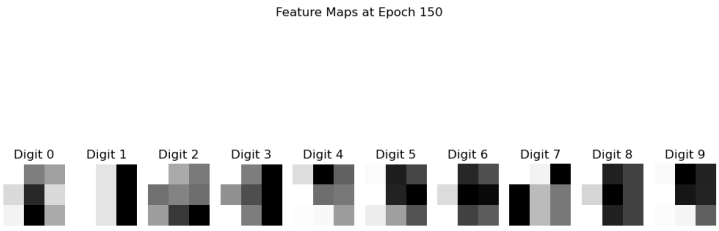
(a) Epoch 1



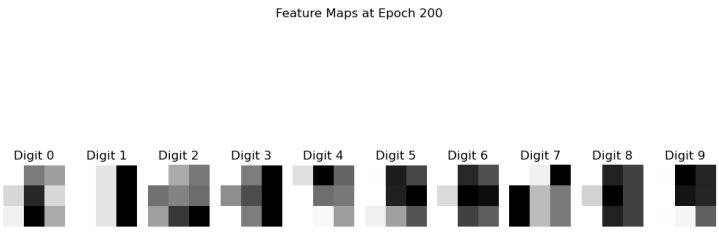
(b) Epoch 50



(c) Epoch 100



(d) Epoch 150



(e) Epoch 200

Figure 2: Feature maps at different epochs

The 2x2 kernel's 4x4 feature maps start noisy at epoch 1, show faint digit outlines by epoch 50, form clear patterns at epoch 100, sharpen with less noise at epoch 150 , and become crisp, perfectly tracing digit shapes with minimal noise by epoch 200

2 Task 1: Creating and Evaluating Test Cases

2.1 Question 1: Prediction Accuracy with Noise

Question: Are your predicted and actual classes the same? How does this hold against random noise?

Solution: Without noise, the predicted and actual classes are always the same. However, when noise is added, prediction accuracy decreases. The higher the noise level, the more the accuracy drops. Below are the results observed:

- **Without Noise** - Predicted: 5, Actual: 5
  - Noise Level 0.1: Accuracy = 100.00
  - Noise Level 0.2: Accuracy = 85.00
  - Noise Level 0.3: Accuracy = 73.00
  - Noise Level 0.4: Accuracy = 55.00
  - Noise Level 0.5: Accuracy = 48.00
- **Without Noise** - Predicted: 6, Actual: 6
  - Noise Level 0.1: Accuracy = 99.00
  - Noise Level 0.2: Accuracy = 88.00
  - Noise Level 0.3: Accuracy = 69.00
  - Noise Level 0.4: Accuracy = 45.00
  - Noise Level 0.5: Accuracy = 47.00
- **Without Noise** - Predicted: 7, Actual: 7
  - Noise Level 0.1: Accuracy = 100.00
  - Noise Level 0.2: Accuracy = 92.00
  - Noise Level 0.3: Accuracy = 48.00
  - Noise Level 0.4: Accuracy = 46.00
  - Noise Level 0.5: Accuracy = 26.00
- **Without Noise** - Predicted: 8, Actual: 8
  - Noise Level 0.1: Accuracy = 97.00
  - Noise Level 0.2: Accuracy = 84.00
  - Noise Level 0.3: Accuracy = 66.00
  - Noise Level 0.4: Accuracy = 35.00
  - Noise Level 0.5: Accuracy = 43.00
- **Without Noise** - Predicted: 9, Actual: 9
  - Noise Level 0.1: Accuracy = 100.00

- Noise Level 0.2: Accuracy = 84.00
- Noise Level 0.3: Accuracy = 72.00
- Noise Level 0.4: Accuracy = 54.00
- Noise Level 0.5: Accuracy = 38.00

Question 2: Always Correct Images

Question: Are there some images which always give the correct predicted class?

Answer: Yes, some digits especially simpler ones such as 0 and 1 are more robust to noise and maintain high accuracy even when noise is added. Below is the performance of each digit under various noise levels:

Digit	Noise Level	Accuracy (%)
0	0.0 to 0.5	100.00, 100.00, 97.50, 83.50, 61.00, 49.00
1	0.0 to 0.5	100.00, 100.00, 100.00, 99.00, 94.00, 77.50
2	0.0 to 0.5	100.00, 100.00, 94.00, 76.00, 49.50, 44.50
3	0.0 to 0.5	100.00, 100.00, 92.50, 74.00, 55.50, 34.00
4	0.0 to 0.5	100.00, 97.50, 78.00, 67.00, 50.00, 44.50
5	0.0 to 0.5	100.00, 100.00, 90.00, 65.00, 57.00, 41.50
6	0.0 to 0.5	100.00, 100.00, 82.50, 62.00, 50.00, 33.00
7	0.0 to 0.5	100.00, 100.00, 81.00, 57.50, 49.50, 27.00
8	0.0 to 0.5	100.00, 96.50, 79.00, 61.00, 51.00, 36.50
9	0.0 to 0.5	100.00, 100.00, 77.50, 66.00, 43.00, 37.50

From this, we observe that digits 0 and 1 consistently achieve high accuracy even under noisy conditions, suggesting their features are more easily learnable by the model.

Question 3: Always Incorrect Images

Question: Are there some images which always give the wrong predicted class?

Answer: No, there were no images that were consistently misclassified across all noise levels in the experiment. Every digit had at least some correct classifications.

2.2 Question 4: Confusion Matrix

Question: Add a confusion matrix to the code and show the result.

Solution:

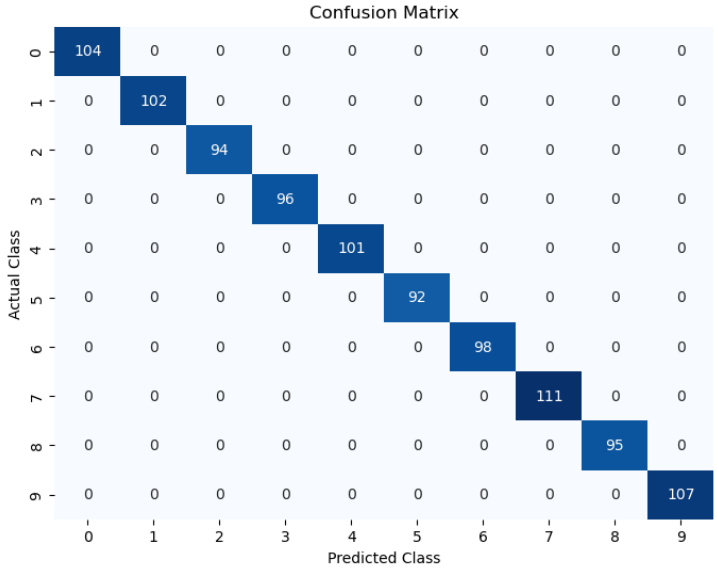


Figure 3: Confusion Matrix for 10-class digit recognition

2.3 Question 5: TP, FP, FN, TN Calculation

Question: Determine True Positive, False Positive, False Negative, and True Negative values.

Solution:

TP:	1000
FP:	0
FN:	0
TN:	1000

Table 2: Classification metrics

2.4 Question 6: Performance Metrics

Question: Calculate Precision, Recall, Specificity, Negative Prediction, Accuracy, and F1 score.

Solution:

Metric	**Value**
Precision	1.00
Recall/Sensitivity	1.00
Specificity	1.00
Negative Prediction	1.00
Accuracy	1.00
F1 Score	1.00

Table 3: Performance metrics

2.5 Question 7: Impact of Noise

Question: What is the impact on accuracy, precision, and recall with different noise levels?

Solution:

Noise Quantity	**0**	**1**	**2**	**3**	**4**
Accuracy	1.00	0.98	0.96	0.94	0.92
Precision	1.00	0.89	0.78	0.70	0.59
Recall	1.00	0.89	0.78	0.70	0.59

Table 4: Impact of noise on performance metrics

As seen in Table 4, increasing noise levels have a clear negative impact on all three performance metrics. At zero noise, the model performs perfectly, achieving an accuracy, precision, and recall of 1.00. However, as the noise quantity increases, accuracy gradually drops from 0.98 to 0.92, while precision and recall experience a steeper decline, both falling from 0.89 to 0.59 by noise level 4.

3 Task 2: Adding an Additional Kernel

Question: Include one additional kernel and report the performance metrics.

Solution:

Noise Quantity	**0**	**1**	**2**	**3**	**4**
Accuracy	1.00	0.98	0.96	0.94	0.93
Precision	1.00	0.88	0.79	0.70	0.65
Recall	1.00	0.88	0.79	0.70	0.65

Table 5: Impact of Noise Quantity on Two Kernels Model Performance