Assignment 1 Part C: Convolutional Neural Networks for Digit Recognition

Deep Learning - EE569

Motaz M Alharbi
2190203271
Instructor: Dr. Nuri Benbarka

November 12, 2025

Contents

1 Introduction

This report details the third phase of Assignment 1, which transitions from Multi-Layer Perceptrons (MLPs) to Convolutional Neural Networks (CNNs). The primary objectives of this part were to implement the core building blocks of CNNs—the convolution and max-pooling layers—within the custom computational graph framework. Subsequently, these layers were used to construct, train, and evaluate CNNs for the task of handwritten digit classification using the scikit-learn ‘digits‘ dataset.

The investigation begins by establishing a baseline performance with a fully-connected MLP. It then documents the implementation and verification of the ‘Conv‘ and ‘MaxPooling‘ nodes. Finally, two CNN architectures—a baseline model and a deeper, optimized model—are trained and analyzed. The performance of these models is systematically compared to demonstrate the effectiveness of CNNs for image-based tasks and to identify the best-performing architecture developed within the framework.

2 Task 1: MLP Baseline on MNIST Digits

2.1 Task Description

To provide a benchmark for evaluating the performance of the CNNs, the first task was to build and train a standard Multi-Layer Perceptron on the same dataset. This establishes a baseline accuracy that a general-purpose neural network can achieve, against which the specialized CNN architectures can be compared.

2.2 Model Architecture

The MLP was constructed using the existing framework components. The input consisted of flattened 8x8 images (64 features). The network architecture is as follows:

$$\text{Input (64x1)} \rightarrow \text{Linear(64, 128)} \rightarrow \text{ReLU} \rightarrow \text{Linear(128, 10)} \rightarrow \text{Softmax}$$

The model was trained for 1000 epochs using full-batch gradient descent with a learning rate of 0.05. The Cross-Entropy loss function was used for optimization.

2.3 Results and Discussion

The MLP model achieved a final test accuracy of **95.00%**. The training loss, shown in Figure 1, demonstrates stable convergence over the 1000 epochs. The confusion matrix in Figure 2 indicates strong performance across all digit classes, with very few misclassifications. Sample predictions (Figure 3) further confirm the model’s high accuracy. While effective, this result serves as the baseline to determine if the spatial feature extraction capabilities of a CNN can provide a performance advantage.

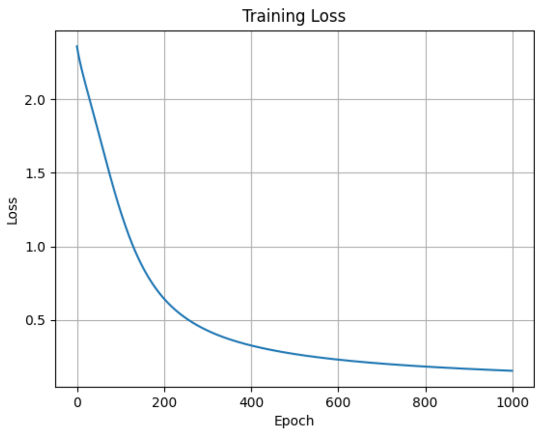


Figure 1: MLP training loss over 1000 epochs.

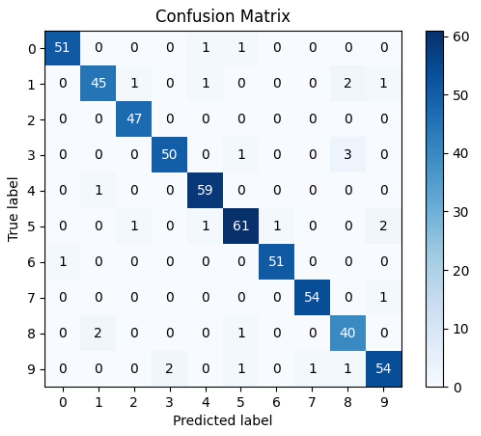


Figure 2: MLP confusion matrix on the test set.

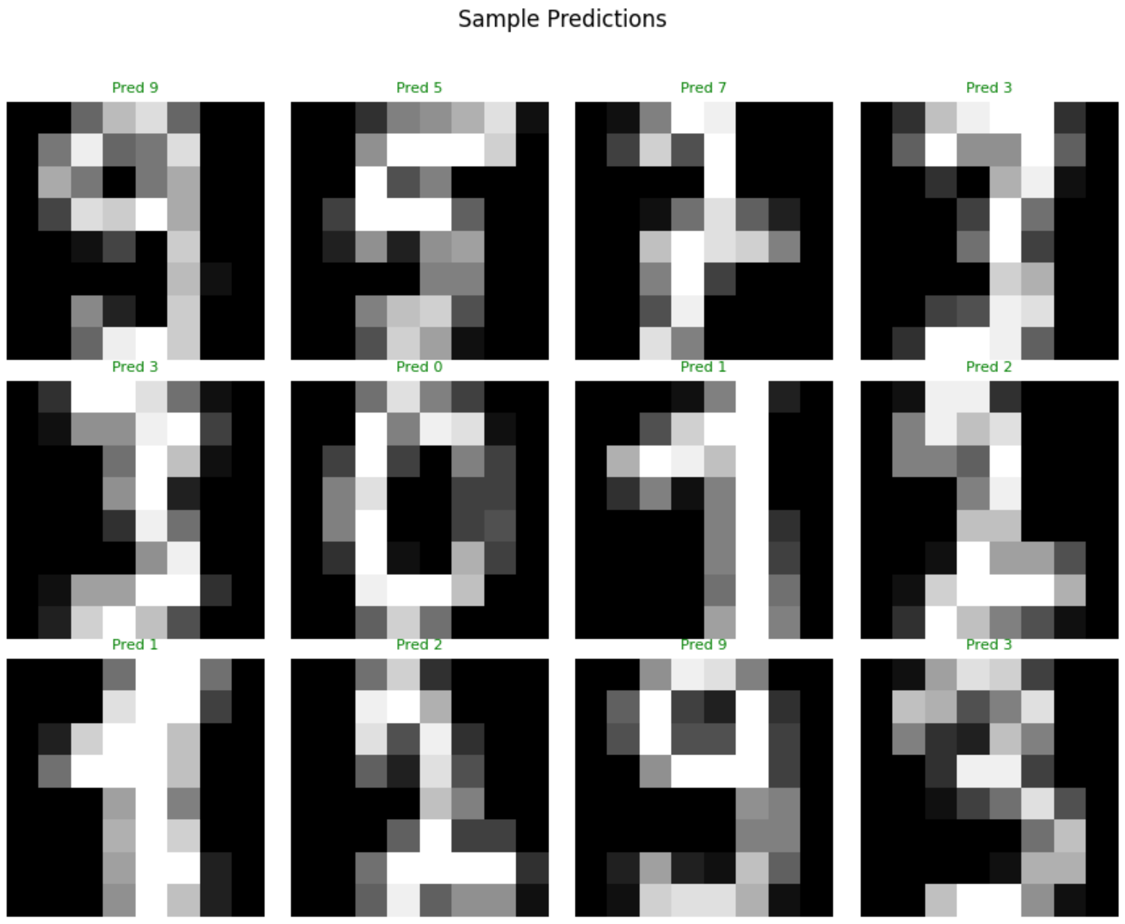


Figure 3: Sample predictions from the trained MLP model.

3 Task 2: Implementing Core CNN Layers

3.1 Task Description

The foundational components of any CNN are the convolution and pooling layers. This task involved implementing ‘Conv‘ and ‘MaxPooling‘ classes as new nodes within the computational graph framework. These nodes must support both a forward pass for computation and a backward pass for gradient propagation.

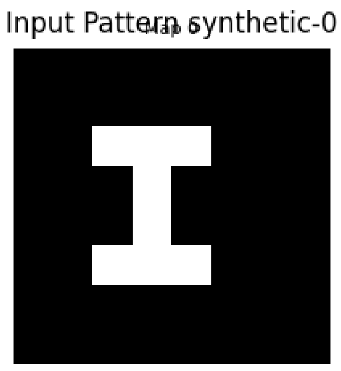
3.2 Implementation Details

Convolution (‘Conv‘) Node: The ‘Conv‘ node was implemented to perform 2D convolution. For efficiency, the forward pass utilizes the ‘im2col‘ technique, which transforms image patches into columns. This allows the convolution operation to be expressed as a single, highly optimized matrix multiplication between the rearranged input and the flattened filter weights. The backward pass calculates the gradients with respect to the input, weights, and biases by reversing this process using ‘col2im‘.

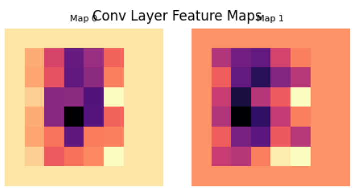
Max Pooling (‘MaxPooling‘) Node: The ‘MaxPooling‘ node is responsible for downsampling the feature maps, reducing their spatial dimensions while retaining the most prominent features. The forward pass identifies the maximum value within each pooling window and records its position. During backpropagation, the gradient is routed exclusively to the location of the maximum value from the forward pass, effectively implementing the sub-gradient of the max operation.

3.3 Verification

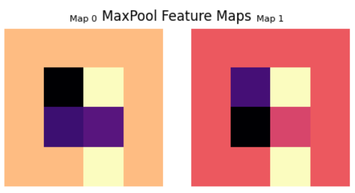
To verify the correct functionality of the new layers, a simple test was conducted using a synthetic 8x8 image pattern. As shown in Figure 4, the input pattern was passed through a ‘Conv‘ layer and then a ‘MaxPooling‘ layer. The resulting feature maps were visualized at each stage, confirming that the layers were transforming the input as expected and producing outputs of the correct shape and content.



Input Pattern



Conv Feature Maps



MaxPool Feature Maps

Figure 4: Verification of ‘Conv‘ and ‘MaxPooling‘ layers on a synthetic input image.

4 Task 3: Constructing a Baseline CNN

4.1 Model Architecture

With the core layers implemented, a baseline CNN was constructed to classify the digits. The architecture was designed to be relatively simple, consisting of two convolutional blocks followed by a fully-connected output layer. ReLU was used as the activation function after each convolution. The architecture is as follows:

1. Input: (Batch, 1, 8, 8)
2. Conv (1→8 filters, 3x3) → ReLU → MaxPool (2x2)
3. Conv (8→16 filters, 3x3) → ReLU → MaxPool (2x2)
4. Flatten → Linear(16*2*2, 10) → Softmax

The model was trained for 50 epochs using mini-batch SGD with a batch size of 32 and a learning rate of 0.01.

4.2 Results and Discussion

The baseline CNN achieved a test accuracy of **95.74%**. This represents a modest improvement over the MLP's 95.00% accuracy. The training loss (Figure 5) shows rapid convergence, stabilizing after approximately 30 epochs. The confusion matrix (Figure 6) and sample predictions (Figure 7) are comparable to the MLP's, demonstrating solid but not overwhelmingly superior performance. This suggests that while the convolutional structure is beneficial, a more complex architecture may be needed to unlock significant performance gains.

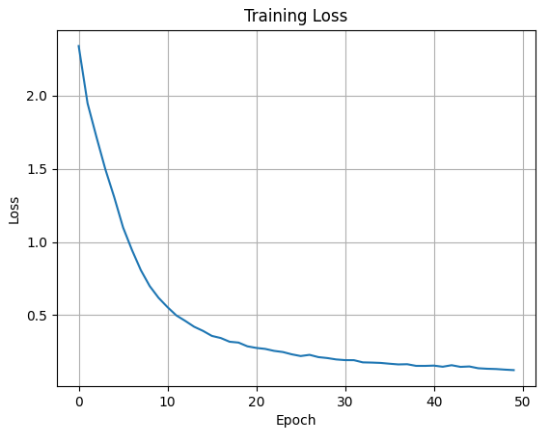
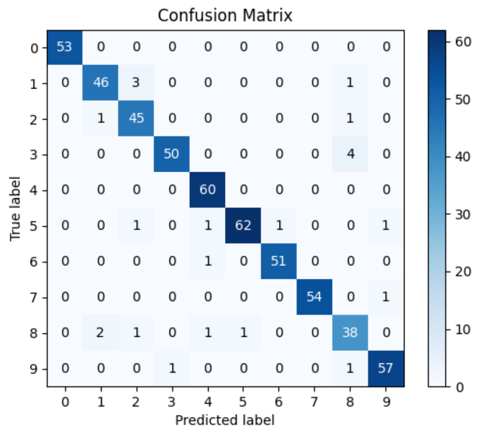


Figure 5: Baseline CNN training loss.



	0	1	2	3	4	5	6	7	8	9
0	53	0	0	0	0	0	0	0	0	0
1	0	46	3	0	0	0	0	0	1	0
2	0	1	45	0	0	0	0	0	1	0
3	0	0	0	50	0	0	0	0	4	0
4	0	0	0	0	60	0	0	0	0	0
5	0	0	1	0	1	62	1	0	0	1
6	0	0	0	0	1	0	51	0	0	0
7	0	0	0	0	0	0	0	54	0	1
8	0	2	1	0	1	1	0	0	38	0
9	0	0	0	1	0	0	0	0	1	57

Figure 6: Baseline CNN confusion matrix.

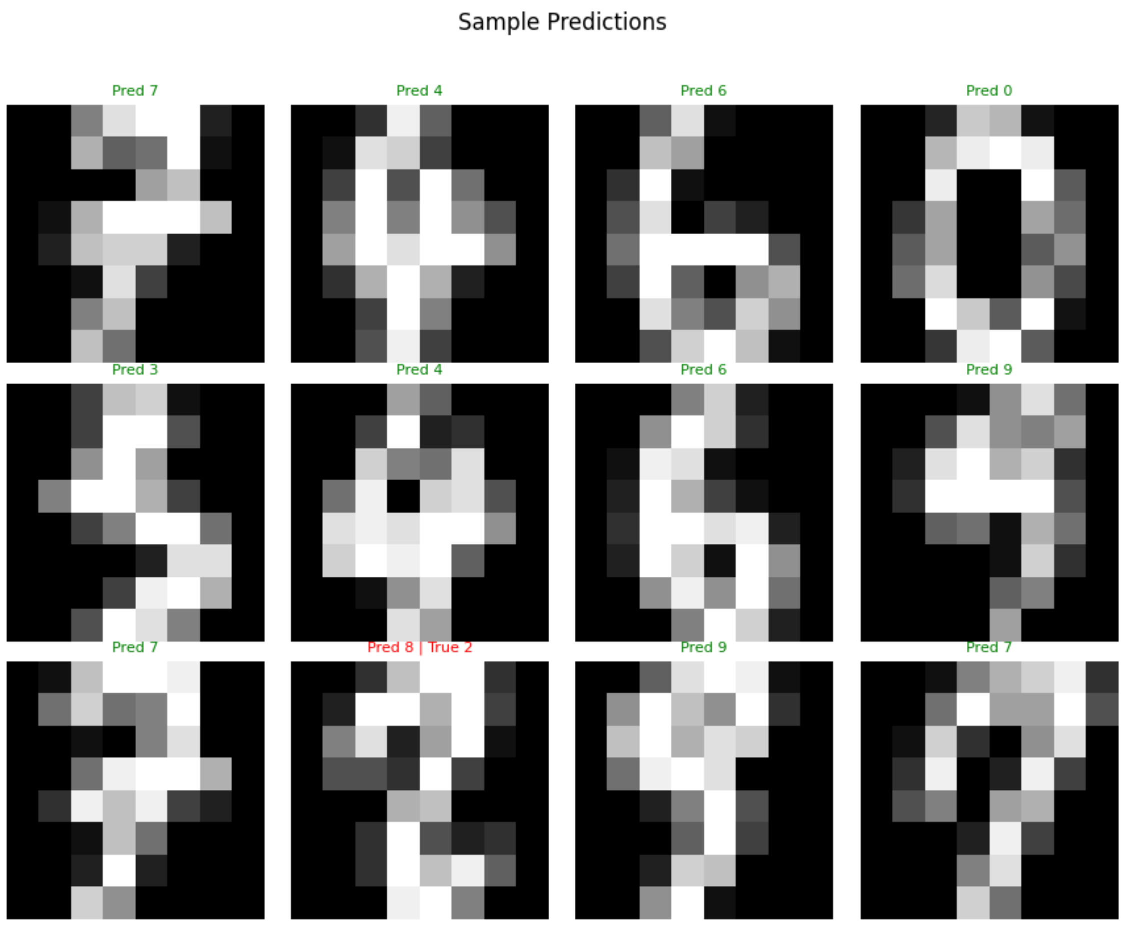


Figure 7: Sample predictions from the baseline CNN model.

5 Task 4: CNN Experimentation and Optimization

5.1 Model Architecture

To achieve the best possible performance, an experiment was conducted with a deeper and wider CNN architecture. This model was designed to learn a more complex hierarchy of features by adding an additional convolutional layer and increasing the number of filters at each stage. The optimized architecture is as follows:

1. Input: (Batch, 1, 8, 8)
2. Conv (1→16 filters, 3x3) → ReLU → MaxPool (2x2)
3. Conv (16→32 filters, 3x3) → ReLU → MaxPool (2x2)
4. Conv (32→64 filters, 3x3) → ReLU → MaxPool (2x2)
5. Flatten → Linear(64*1*1, 10) → Softmax

This model was trained for 80 epochs with a batch size of 32 and a learning rate of 0.02.

5.2 Results and Discussion

The deeper CNN architecture yielded a significant performance improvement, achieving a final test accuracy of **98.33%**. This result is substantially better than both the MLP (95.00%) and the baseline CNN (95.74%). The loss curve (Figure 8) shows smooth convergence, reaching a very low value. The confusion matrix (Figure 9) is nearly diagonal, indicating extremely high per-class accuracy. This superior performance demonstrates the power of hierarchical feature learning in deeper CNNs, where initial layers learn simple features like edges, and subsequent layers combine them to recognize more complex patterns.

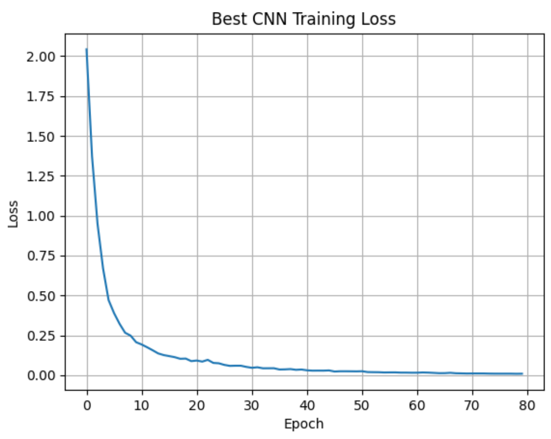
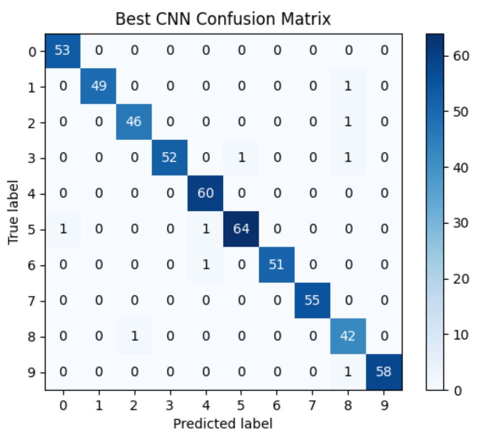


Figure 8: Optimized CNN training loss.



True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	53	0	0	0	0	0	0	0	0	0
1	0	49	0	0	0	0	0	0	1	0
2	0	0	46	0	0	0	0	0	1	0
3	0	0	0	52	0	1	0	0	1	0
4	0	0	0	0	60	0	0	0	0	0
5	1	0	0	0	1	64	0	0	0	0
6	0	0	0	0	1	0	51	0	0	0
7	0	0	0	0	0	0	0	55	0	0
8	0	0	1	0	0	0	0	0	42	0
9	0	0	0	0	0	0	0	0	1	58

Figure 9: Optimized CNN confusion matrix.

5.3 Overall Comparison

The results from all three models are summarized in Table 1 and visualized in Figure 11. The progression is clear: while the MLP is a strong general model, the CNN architecture is inherently better suited for image data. Furthermore, increasing the depth and capacity of the CNN allowed it to learn more discriminative features, leading to the highest accuracy.

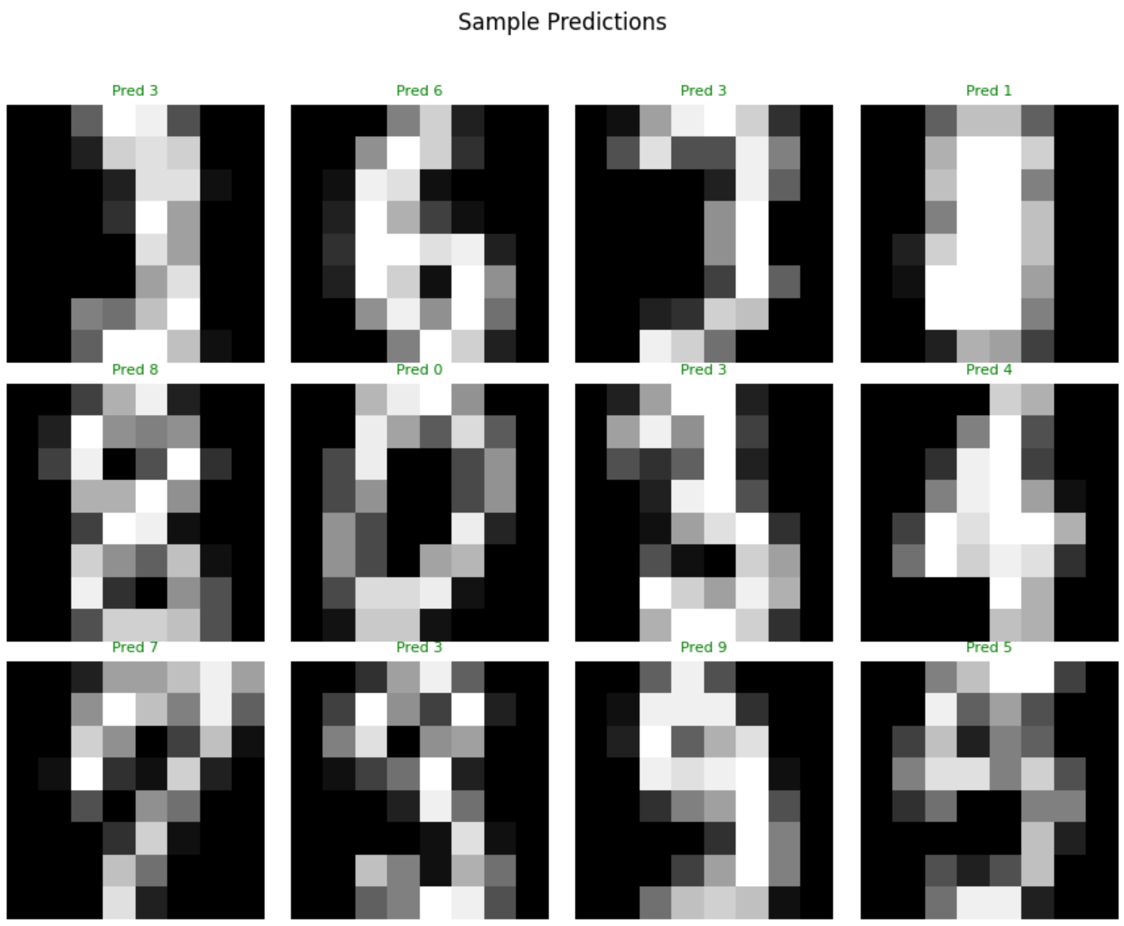


Figure 10: Sample predictions from the optimized CNN model.

Table 1: Performance Comparison of All Models

Model	**Test Accuracy (%)**	**Macro F1 Score**
MLP (Task 1)	95.00	0.950
Baseline CNN (Task 3)	95.74	0.955
Optimized CNN (Task 4)	**98.33**	**0.983**

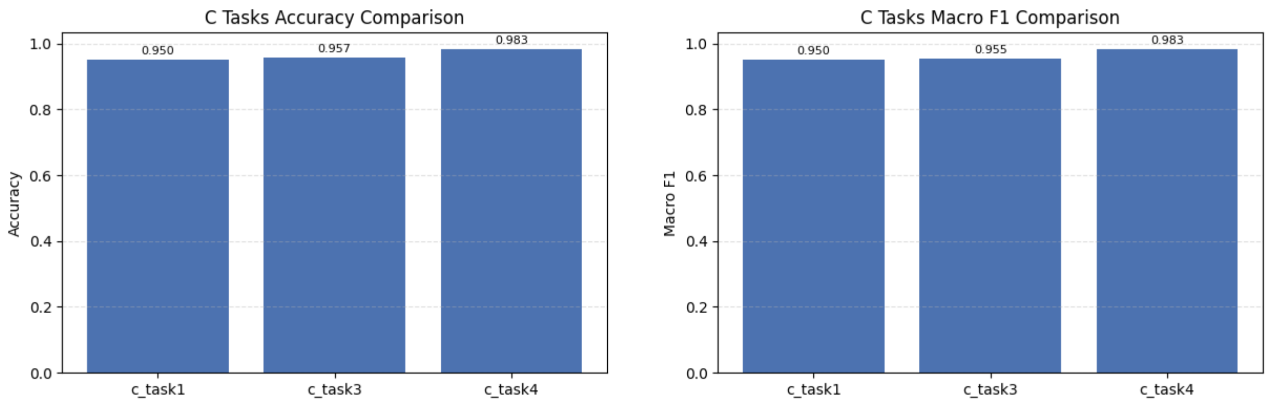


Figure 11: Summary comparison of final model performance: Accuracy (left) and Macro F1 Score (right).

6 Conclusion

This assignment successfully demonstrated the process of building, training, and evaluating convolutional neural networks from the ground up within a custom framework. The core ‘Conv‘ and ‘MaxPooling‘ layers were implemented and verified, enabling the construction of effective models for image classification.

The empirical results unequivocally confirmed the superiority of CNNs over MLPs for this task. The optimized, deeper CNN architecture achieved a test accuracy of **98.33%**, significantly outperforming the MLP baseline of 95.00%. This highlights the critical advantage of leveraging spatial hierarchies and parameter sharing through convolutional layers for processing image data. The project serves as a practical validation of fundamental deep learning principles for computer vision.