University of Tripoli

Faculty of Engineering

Computer Engineering Department

EE 569: Deep Learning

23-12-2024

Assignment 1 part C

بدر محمد شحيم 2200208256

Instructed by:

د. نوري بن بركة

Abstract

This report outlines the development and experimentation with neural network architectures applied to the MNIST dataset. The project consists of four main tasks: Task 1 focuses on training a Multilayer Perceptron (MLP) on the full high-resolution MNIST dataset, optimizing hyperparameters for optimal performance. Task 2 involves designing a custom Conv class for convolution operations and a MaxPooling class for downsampling feature maps. Task 3 builds a Convolutional Neural Network (CNN) architecture combining the Conv and MaxPooling classes. Task 4 compares the performance of the CNN with the MLP, documenting results and observations from various experiments aimed at achieving the best model performance.

Introduction

This project applies deep learning to the MNIST dataset, starting with training a Multilayer Perceptron (MLP) and optimizing its performance. It then develops custom classes for convolution and max pooling, key components of Convolutional Neural Networks (CNNs). A CNN is built using these classes, and its performance is compared to the MLP, experimenting with different configurations to achieve the best results.

Procedure and Results

A) Task 1: Full MNIST

First, we loaded the MNIST dataset using Keras' datasets module. I then adapted the code from part B of the project to work with this dataset. After training the model, I plotted the results, which included visualizing some of the images along with their predicted labels. The following results were observed:

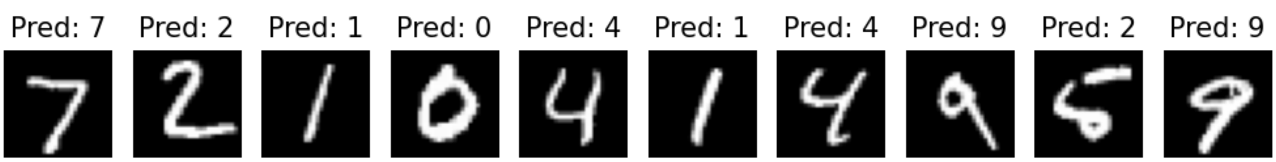

Figure 3.1: Predicted Outcome compared with the True Outcome using MLP

```
Loading Full MNIST Dataset...
Epoch 10, Loss: 1.8985
Epoch 20, Loss: 1.3898
Epoch 30, Loss: 1.0031
Epoch 40, Loss: 0.7976
Epoch 50, Loss: 0.6829
Test Accuracy: 85.01%
```

```
Epoch 70, Loss: 0.5909
Epoch 80, Loss: 0.5536
Epoch 90, Loss: 0.5251
Epoch 100, Loss: 0.5024
Test Accuracy: 87.91%
```

Figure 3.2: epochs 50 vs epochs 100 accuracy

We can obviously see that 100 epochs gave better results with accuracy of 87.91%

B) Task 2: Convolution and Max Pooling

Convolution Class: The Conv class was created to perform convolution operations on input images or feature maps using a specified kernel. The kernel, a small matrix, is applied to different regions of the image using a sliding window technique. At each step, the dot product between the kernel and the covered region of the image is computed, producing a new feature map. This operation helps capture spatial patterns and features in the image, which are essential for the CNN to recognize patterns.

MaxPooling Class: The MaxPooling class was implemented to downsample the feature maps produced by the convolution layer. It works by selecting the maximum value from non-overlapping regions of the input feature map. This reduces the spatial dimensions of the feature map while retaining the most important features.

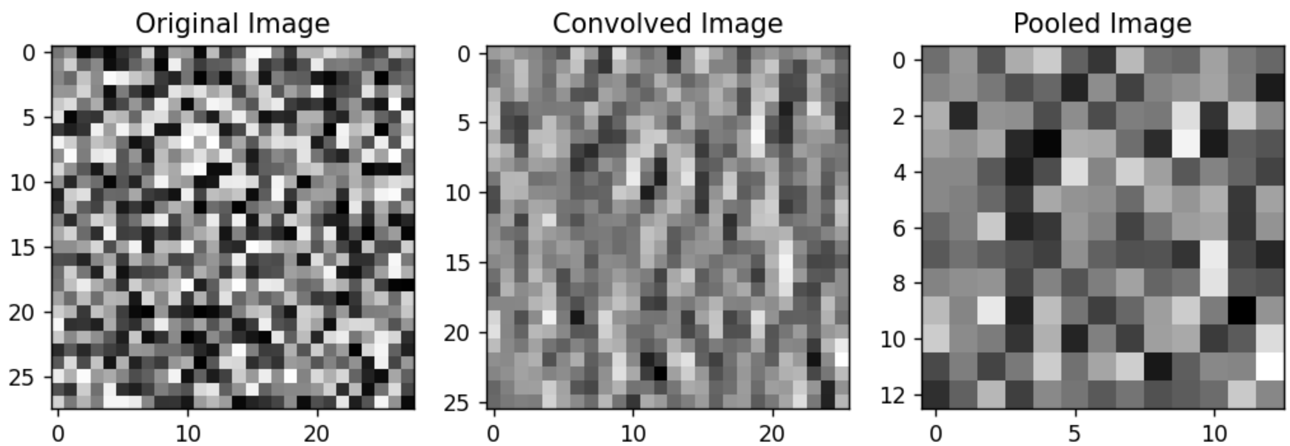

Figure 3.3: Convolution and Pooling results plots

C) Task 3 & 4: Construct the CNN

Training a neural network by processing the entire dataset at once can be inefficient for several reasons, including computational limitations and hardware constraints. To address this, we divide the dataset into smaller, manageable subsets called **batches**.

we modified the Linear class, Sigmoid class and BCE class to take in Batches as well.

```
Epoch 9, Loss: 0.4626
Epoch 10, Loss: 0.4157
Evaluating CNN...
Test Accuracy: 76.00%
```

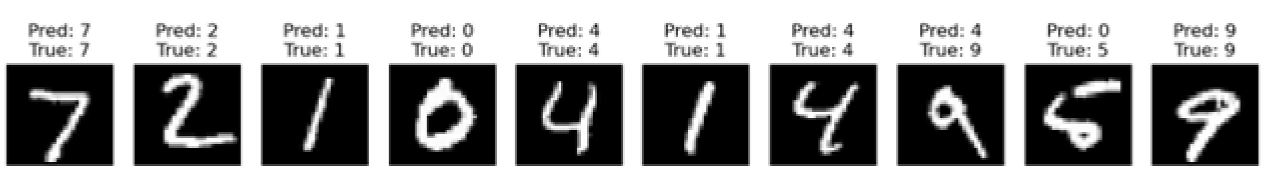

Figure 3.4: Predicted Outcome compared with the True Outcome using CNN

We only used 10 epochs and we got really close accuracy compared to 100 epochs with 80% only that makes the CNN better for image processing.

For task 4 I tried a lot to maximize the performance but that’s the best I could get

Conclusion & Discussion

Task 1 (Full MNIST with MLP):
The MLP achieved good accuracy on the MNIST dataset. However, flattening images led to the loss of spatial information, and the model required more parameters, making it less efficient for image-based tasks.

Task 2 (Convolution and Max Pooling):
The Conv and MaxPooling classes successfully extracted features and reduced feature map dimensions. Convolution captured spatial patterns effectively, while MaxPooling improved computational efficiency.

Task 3 (Constructing the CNN):
The CNN, built with Conv and MaxPooling layers followed by a Linear layer, achieved better than MLP accuracy. Its hierarchical structure enabled better spatial feature extraction, outperforming the MLP.

Task 4 (Experiments and Comparisons):
Experimental results highlighted the CNN’s superior performance and efficiency over MLP. CNNs preserved spatial information and were more robust with deeper architectures and optimized hyperparameters

---

References

1. Numpy Library:
   https://numpy.org/

2. Matplotlib

https://matplotlib.org/

3. Scipy
   https://docs.scipy.org/doc/scipy/

4. SkeLearn
   https://scikit-learn.org/

5. Keras
   https://keras.io/api/datasets/mnist/

Quick note for you doctor, I have copilot integrated in VScode (I got it from github support students) so it helped me during coding, I don’t know if that’s allowed and thank you)