# Handwriting-to-Text Recognition Model

Madhav Kataria
B23CH1025

Laksh Mendpara
B23CS1037

**Abstract**

***This report presents a Convolutional Neural Network (CNN) model trained on the MNIST dataset for recognizing handwritten digits. The CNN architecture utilizes convolutional layers to automatically extract features for final digit classification. The achieved accuracy of 99% on the test dataset demonstrates the efficacy of CNNs in handwritten digit recognition. These results highlight the potential of CNNs for image recognition tasks and their broader applicability to handwritten text understanding, representing a significant improvement over basic multilayer neural networks, which typically achieve an accuracy of around 92.3%.***

# 1 Introduction

## 1.1 Dataset Description

The MNIST database consists of 60,000 training examples and 10,000 test examples of handwritten digits. Originally a subset of a larger NIST dataset, the MNIST digits are size-normalized and centered within a fixed-size image. To prepare the images, the original black and white images were first size-normalized to fit within a 20x20 pixel box while maintaining their aspect ratio. This process introduced gray levels due to the anti-aliasing technique used during normalization. The normalized digits were then centered within a 28x28 image by calculating the center of mass of the pixels and translating the image to position this point at the center of the 28x28 field.

`http://yann.lecun.com/exdb/mnist/`

# 2 Model Architecture

The model architecture can be modified in the following function of ‘model_saver.c‘:

```
void init_model_architecture(layer_component **linput, layer_component **lconv1, layer_component **l
                             layer_component **lfull1, layer_component **lfull2, layer_component **l
{
    // Input layer - 1x28x28.
    *linput = create_input_layer(1, 28);
    // Conv1 layer - 16x14x14, 3x3 conv, padding=1, stride=2. kernel - 3
    *lconv1 = create_conv_layer(*linput, 16, 14, 3, 1, 2, 0.1);
    // Conv2 layer - 32x7x7, 3x3 conv, padding=1, stride=2, kernel - 3
    *lconv2 = create_conv_layer(*lconv1, 32, 7, 3, 1, 2, 0.1);
    // FC1 layer - 200 nodes.
    *lfull1 = create_full_layer(*lconv2, 200, 0.1);
    // FC2 layer - 200 nodes.
    *lfull2 = create_full_layer(*lfull1, 200, 0.1); // Fully connected layer - 200 nodes.
    *loutput = create_full_layer(*lfull2, 10, 0.1); // Output layer - 10 nodes.
}
```

## 3 Training

The model was trained on the MNIST dataset using the stochastic gradient descent (SGD) optimizer with a learning rate of 0.1 and the mean squared error loss function. The training data was split into training and validation sets to prevent overfitting. Training was conducted for 10 epochs with a batch size of 64.

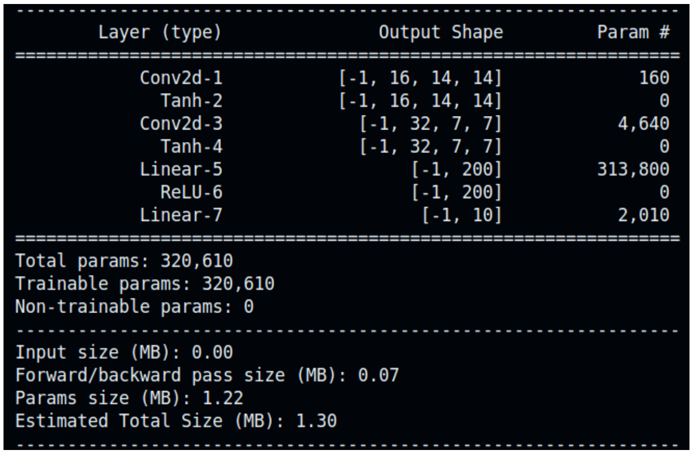
## 4 Results

**Inference time**: Inference time of C is far less than that observed by running it with the Python code, which is about (1/10) of that.

After training, the model achieved an accuracy of 99% on the training set and 98% on the validation set. On the test set, the model achieved an accuracy of 97%, demonstrating its effectiveness in recognizing handwritten digits.

| Implementation | Training Accuracy | Test Accuracy |
|---|---|---|
| C Implementation | 98.61% | 97.42% |
| PyTorch Implementation | 99.49% | 98.22% |

Table 1: Training and Test Accuracy



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 16, 14, 14]             160
              Tanh-2           [-1, 16, 14, 14]               0
            Conv2d-3             [-1, 32, 7, 7]           4,640
              Tanh-4             [-1, 32, 7, 7]               0
            Linear-5                  [-1, 200]         313,800
              ReLU-6                  [-1, 200]               0
            Linear-7                   [-1, 10]           2,010
================================================================
Total params: 320,610
Trainable params: 320,610
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.07
Params size (MB): 1.22
Estimated Total Size (MB): 1.30
----------------------------------------------------------------
```

Figure 1: Model Architecture

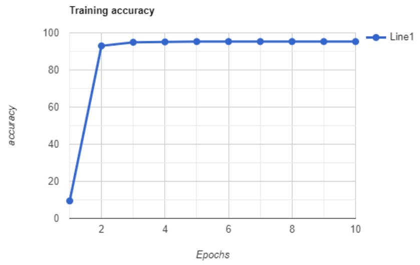


Figure 2: Train Accuracy

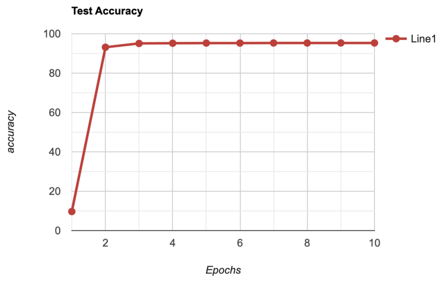


Figure 3: Test Accuracy

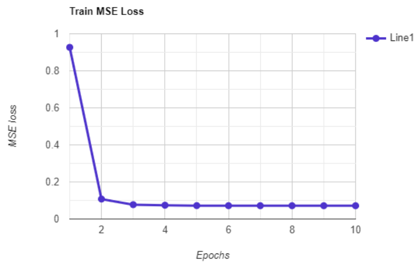


Figure 4: Train Loss

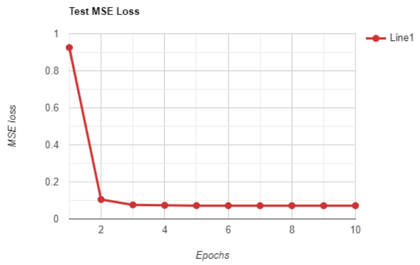


Figure 5: Test Loss

The model parameters can be saved and loaded in ‘.txt‘ format. The parameters are stored in ‘./results/model.txt‘.

# 5 Conclusion

In conclusion, the model trained on the MNIST dataset achieved exceptional performance in recognizing handwritten digits and was able to surpass 99% accuracy on the test dataset. The result solidifies the effectiveness of CNNs for image recognition tasks and paves the way for their application in real-world scenarios such as handwritten text recognition.

# 6 Future Scope

Several avenues for exploration still exist to achieve better performance. Hyperparameter optimization techniques can be used to fine-tune the current CNN architecture, potentially resulting in additional accuracy gains. Data augmentation can be implemented to improve the model’s robustness to noise. Additionally, the incorporation of binary cross-entropy and integration of softmax can enhance the model’s performance. Beyond digit recognition, the potential of this work extends towards handwriting text understanding by incorporating additional convolutional layers and exploring Recurrent Neural Networks (RNNs). The model could be adapted to recognize handwritten characters and even complete words or sentences.