# **OCR-Based Hand Written Digit Recognition**

-Bhupesh P S -9994615335 -[bhupeshps2003@gmail.com](mailto:bhupeshps2003@gmail.com)

-<https://github.com/bhupesh-p-s/OCR-Based-Hand-Written-Digit-Recognition>

## **1. Introduction**

This project focuses on recognizing digit sequences from image files using Optical Character Recognition (OCR) techniques combined with deep learning. The primary goal was to accurately extract and classify digit sequences from image data, demonstrating a strong understanding of OCR and deep learning concepts.

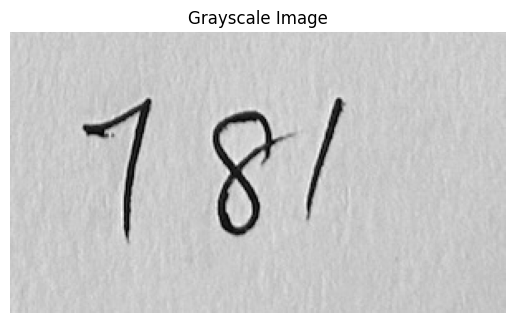
## **2. Approach**

#### **1. Initial Attempt and Challenges**

* Trained the model on the MNIST dataset without preprocessing, leading to the entire sequence being treated as a single digit (predicted 8 for all sequences).
* Model couldn't distinguish individual digits, highlighting the need for segmentation.

#### **2. Preprocessing for Sequence Segmentation**

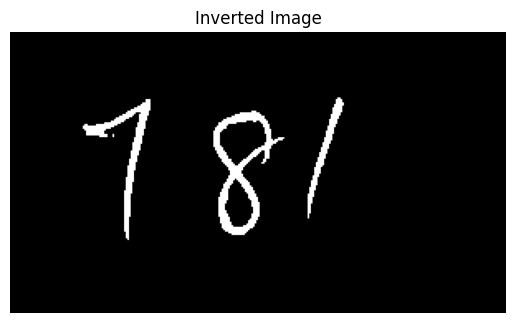
* **Grayscale Conversion**: The input image is loaded in grayscale for better contrast and clarity.



* **Binarization**: Otsu's thresholding method is applied to binarize the image, converting it into a black-and-white image for easier processing.

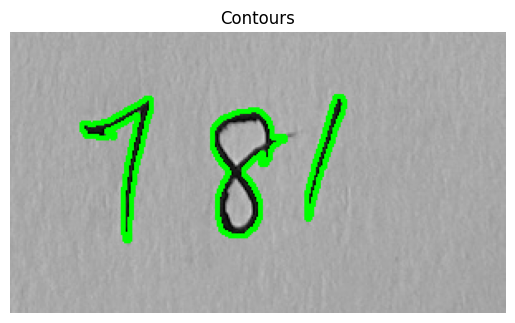


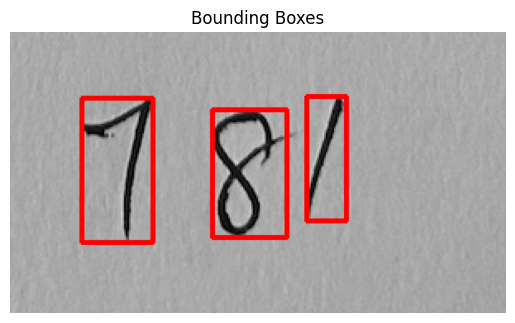
* **Inversion**: The binary image is inverted, as white digits on a black background make it easier for the model to identify the shapes.



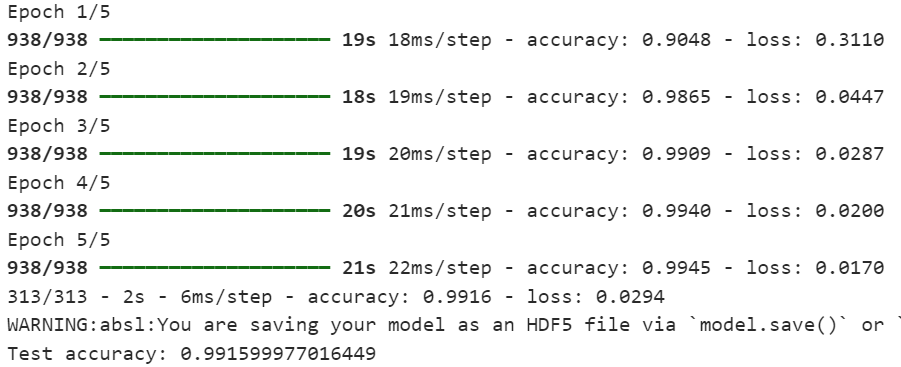
* **Dilation**: A kernel is defined for dilation to enhance the digits and make the contours clearer. The image is dilated to emphasize the digits, making them easier to detect by contour analysis.



* **Contour Extraction**: Contours are detected to locate areas where the digits appear in the image.  
    
  
* **Bounding Boxes:** Bounding boxes are created by iterating through the contours to segment each digit in the sequence into its own.

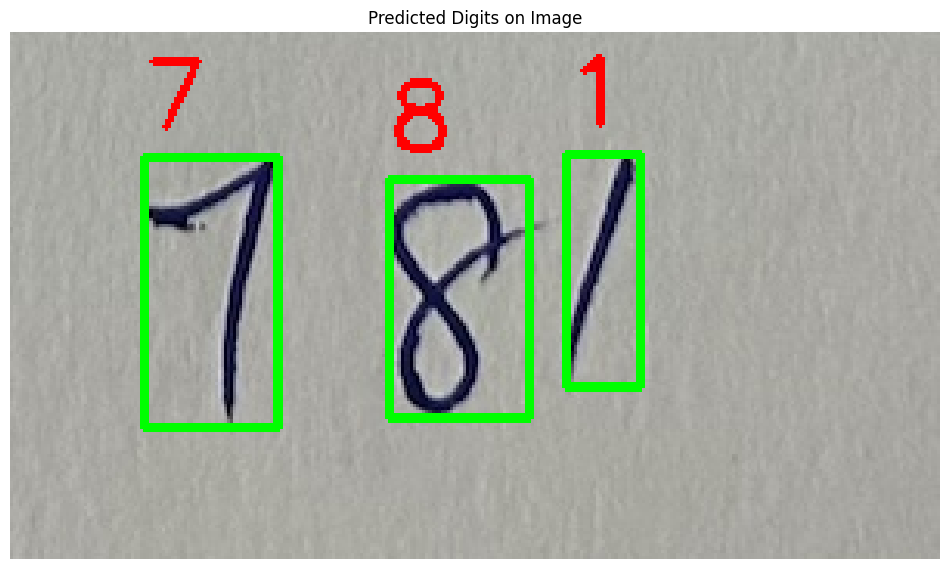


#### **3. Model Selection: CRNN (Convolutional Recurrent Neural Network)**

* **CNN Layers**: Extracted spatial features from images (edges, shapes).
* **LSTM Layers**: Captured sequential dependencies between digits in a sequence.
* **CTC Loss**: Allowed the model to learn without needing explicit alignment of sequence labels, ideal for OCR tasks with variable-length sequences.
* **Convolutional Layers**: 3 layers (32, 64, 128 filters) for progressively abstracting image features.
* **LSTM Layers**: 2 layers with 128 units each to process sequence relationships.
* **Fully Connected Layer**: For final digit prediction.
* **Output Layer**: Softmax for multi-class classification of digits.  
  
* This model gave an **accuracy of** **99.15%** while training and testing it with the **MNIST data set**.

#### **3.Digit Recognition:**

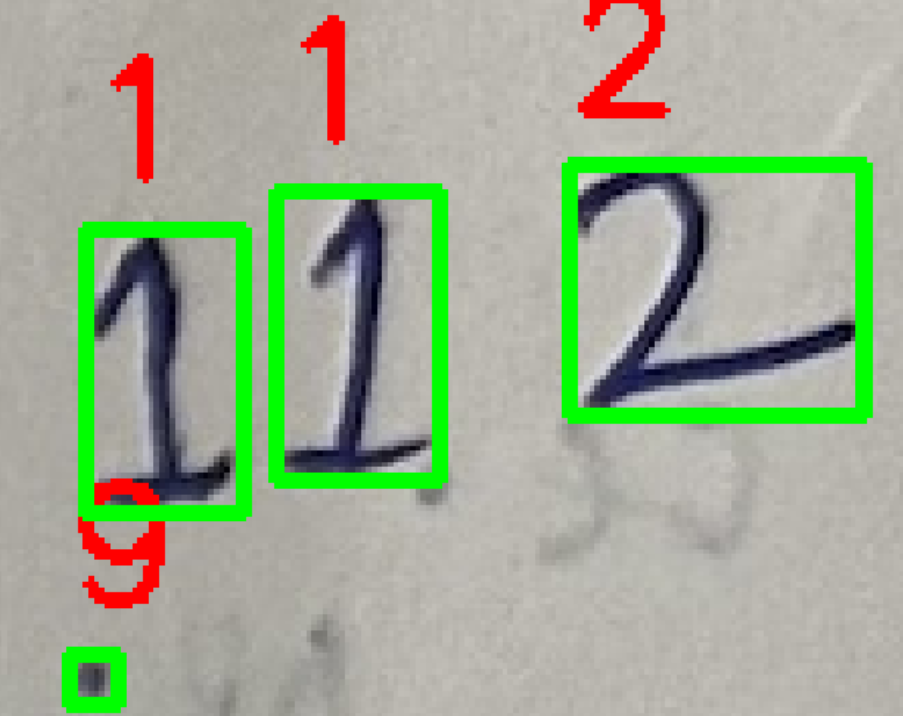
* The pre-trained model is used to predict the digit from the processed image. The output is then mapped to the corresponding digit.

.

## 

## **3. Challenges Faced**

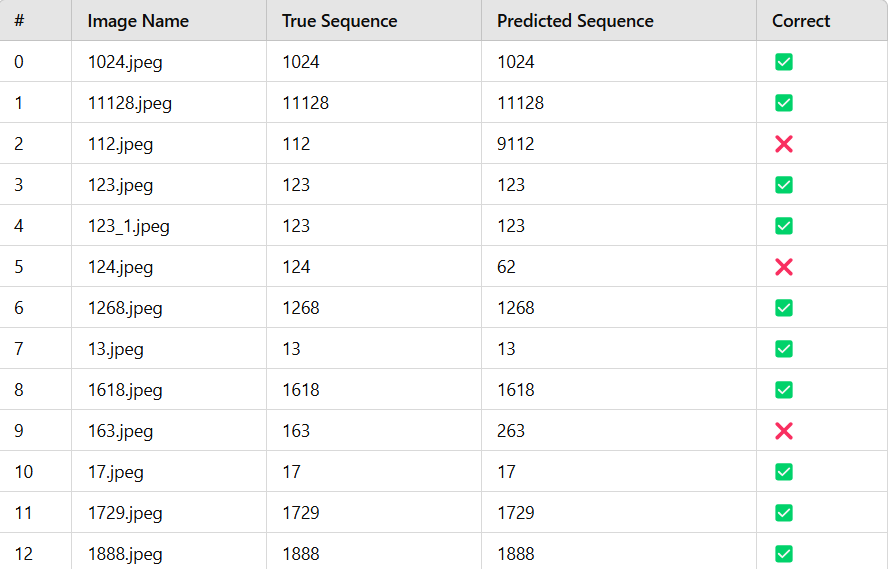
* **Model Misclassification**: Initially, without pre-processing, the model treated the entire sequence as a single digit and predicted "8" for all sequences. Which was rectified using proper preprocessing and segmentation.
* **Handling Noise**: Small dots and noise were mistakenly classified as the digit "9". This was resolved by filtering out small contours to improve segmentation.

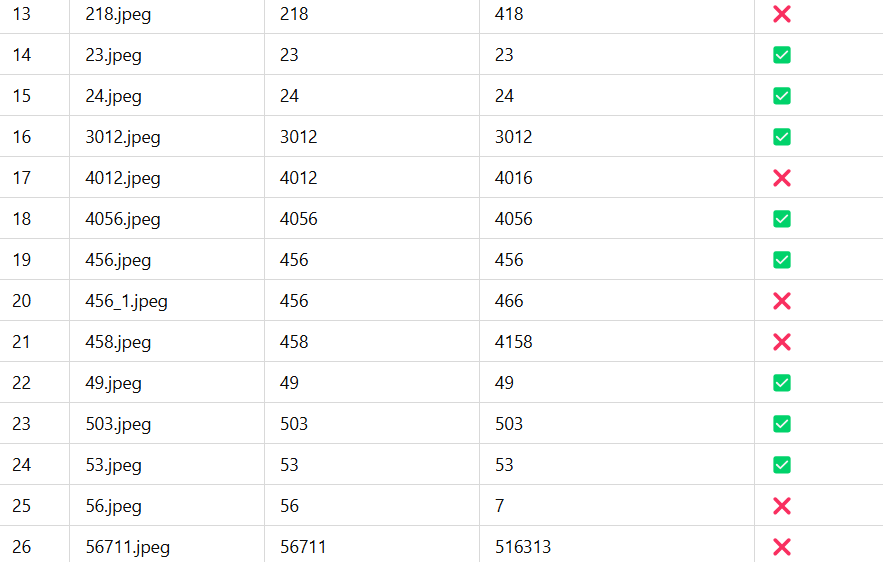


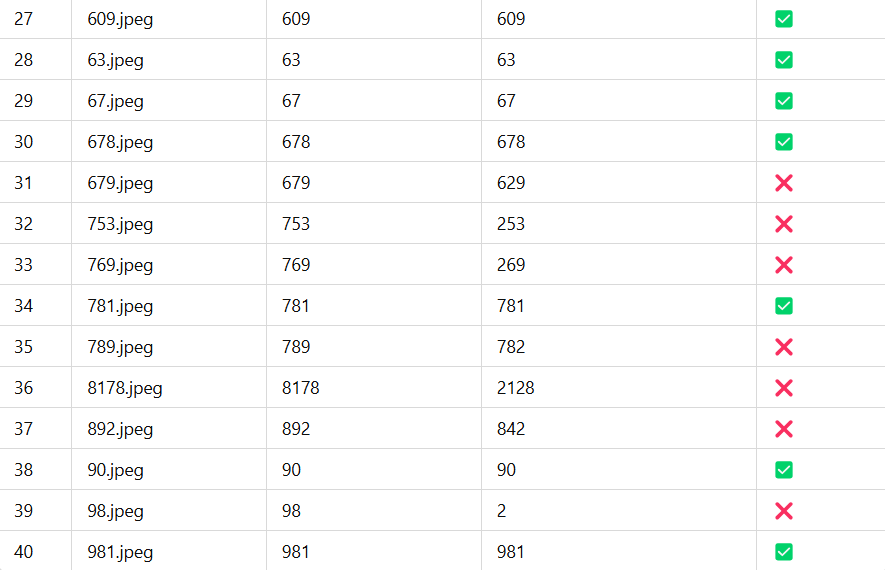
## **4. Observations and Results**

### **Extracted Sequences**

The table below lists the image file names along with the corresponding digit sequences identified during the process.







### **Accuracy**

* Total Predictions: **41**
* Total Correct Predictions: **25**

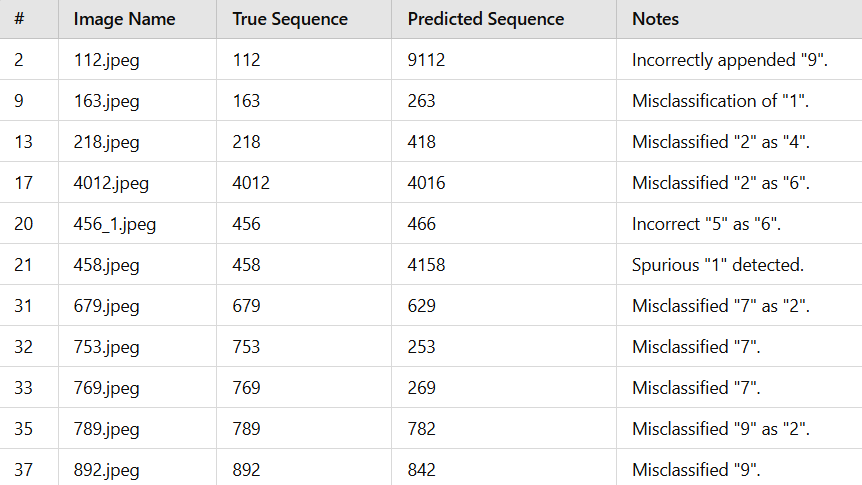
### 

### 

### 

### **Mismatched Predictions - Split**

#### **Predicted Closely**



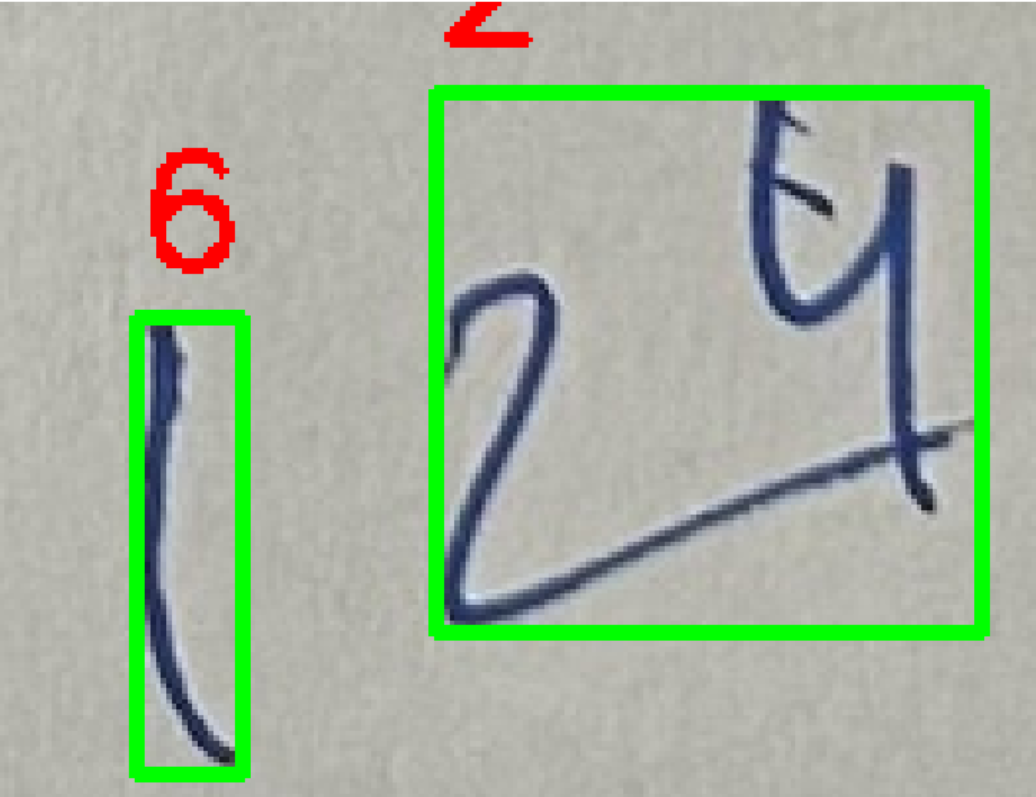
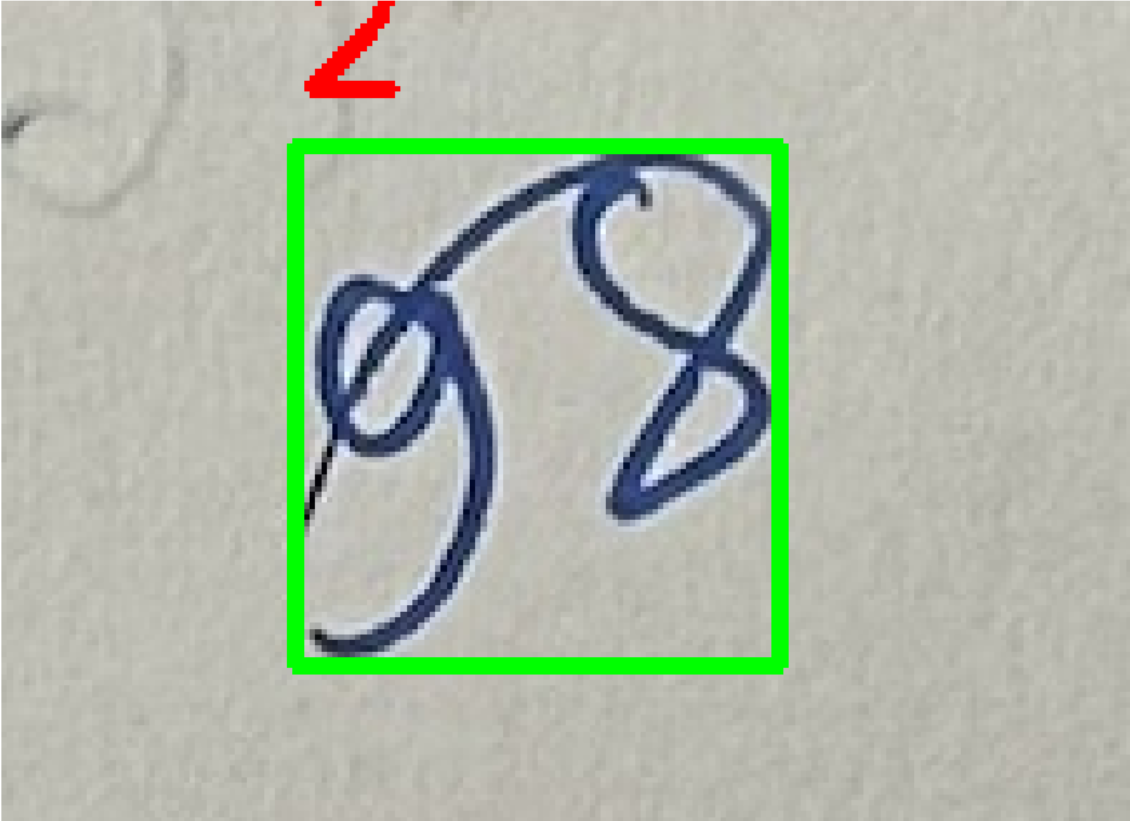
#### **Completely Mismatched**

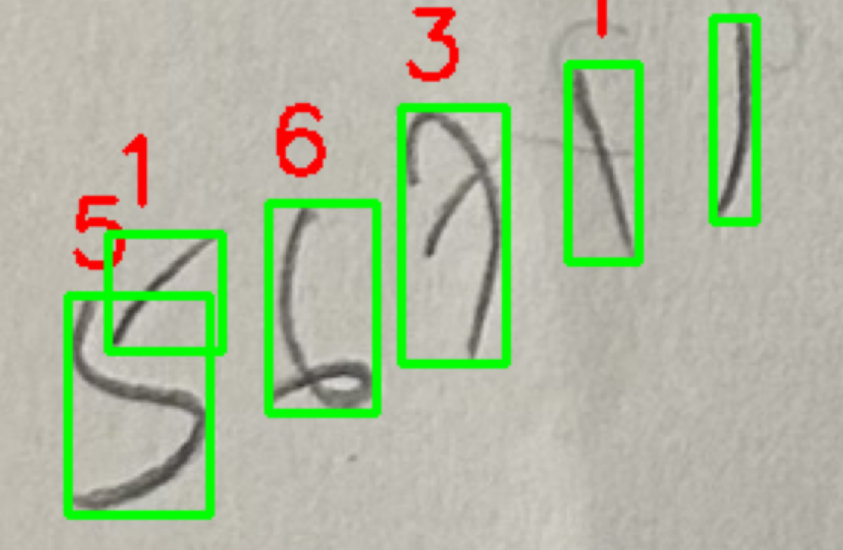
#### 

### **6. Accuracy of the model for the give data:**

* This accuracy was calculated by using a **ground truth table** created having one column as the file name and another column as the true values.
* A python code compares each of the predicted values with the ground truth table to calculate the accuracy.
* The model was able to predict **25 images correctly out of the 41 images**.
* The model was able to **predict 11 digits with only variation in one digit** and noise getting considered as an additional digit.
* Only **5 of the values where totally mispredicted** due to combine writing of the digits.
* Which gave us an **accuracy of 60.98%** for the data of 41 images.

## **7. Uncertainty in some digits:**

* Some digits had written in a combined manner which was not able to be properly segmented and was considered a single digit.  
    
  
* The model predicted certain digits as some other digits like 7 as 3, 1 as 2 etc. as the written format was not clear.



## 

## **8. Conclusion**

Applied **OCR** and **deep learning** to predict sequences of handwritten digits. The process involved:

* **Preprocessing**: Techniques like **binarization**, **image inversion**, and **dilation** were used to enhance digits and filter out noise.
* **Model**: A **CNN** was trained to recognize individual digits, using several convolutional and pooling layers followed by fully connected layers for classification.

The model correctly predicted 25 out of 41 images. While it handled most sequences well, challenges like noise and overlapping digits led to some mispredictions. Future improvements could include better noise handling and digit separation.