PREMIER UNIVERSITY, CHATTOGRAM

Department of Computer Science & Engineering

Pattern Recognition Laboratory(CSE 460)

Final Project Report

On

DIGIT RECOGNITION USING CNN

SUBMITTED BY

Name: Ifthekar Hossain

ID: 1903610201801

Name: Abidur Rahman Bhuiyan

ID: 1903610201814

Name: Md Hafizul Islam

ID: 1903610201822

SUBMITTED TO

Mrs. Noortaz Rezoana

Lecturer

Department of Computer Science & Engineering

Premier University, Chattogram

23 September, 2023

Table of Contents

Abstract

Digit recognition is a fundamental problem in computer vision and pattern recognition with numerous applications, including optical character recognition (OCR), automated mail sorting, and digit-based authentication. This report presents a comprehensive study on the application of Convolutional Neural Networks (CNNs) for accurate and efficient digit recognition.The primary objective of this research is to design and implement a CNN-based model capable of recognizing handwritten digits from the MNIST dataset, a benchmark dataset in the field. The report discusses the model architecture, training process, hyperparameter tuning, and performance evaluation. We also explore techniques such as learning rate scheduling to enhance model performance.The experimental results showcase the model's ability to achieve high accuracy in digit recognition while highlighting the importance of hyperparameter tuning and regularization. Furthermore, we analyze the confusion matrix and present a classification report to assess the model's precision, recall, and F1-score for each digit class. Overall, this research provides valuable insights into the development of CNN models for digit recognition and highlights best practices for achieving accurate and robust results in this critical domain.

Keywords: Digit Recognition,Convolutional Neural Networks (CNNs),MNIST Dataset, Computer Vision,Pattern Recognition,Handwritten Digit Recognition,Optical Character Recognition (OCR),Machine Learning,Learning Rate Scheduling,Model Evaluation,Classification Report,Hyperparameter Tuning,Image Processing,Neural Network Architecture.

CHAPTER 1

INTRODUCTION

1.1 Introduction

In an era characterized by the proliferation of digital data, the ability to recognize and interpret numerical symbols plays a pivotal role in numerous applications across various domains. From automated postal sorting systems to digit-based authentication methods, the accurate recognition of digits is a fundamental task in the field of computer vision and pattern recognition. The quest for precise and efficient digit recognition has fueled extensive research efforts, leading to the development of sophisticated machine learning algorithms and neural network architectures. This report delves into the realm of digit recognition, focusing on the application of Convolutional Neural Networks (CNNs), a class of deep learning models that have demonstrated exceptional prowess in image-related tasks. Specifically, we explore the use of CNNs to recognize handwritten digits, a problem that has served as a benchmark in the machine learning community for decades.Digit recognition is more than just a technical endeavor; it carries significant practical implications. Optical Character Recognition (OCR) systems heavily rely on digit recognition to process printed text, improving document digitization and enabling efficient text extraction. Additionally, applications in the finance sector, such as check processing and numerical data extraction, rely on accurate digit recognition.

1.2 Motivation

Digit recognition has emerged as a foundational challenge in the realm of computer vision and pattern recognition, with profound implications across various industries and applications. As the world becomes increasingly digital and data-driven, the demand for accurate and efficient methods to interpret numerical symbols has never been more pressing. The motivation behind this research is driven by several key factors:

1. Ubiquity of Handwritten and Printed Digits: Handwritten and printed digits are pervasive in our daily lives, appearing on documents, checks, invoices, and a myriad of other sources. Accurate recognition of these numerical symbols is critical for tasks ranging from financial data processing to automated document analysis. A robust digit recognition system can streamline these processes, reduce errors, and enhance productivity.

2. Advancements in Deep Learning: The resurgence of deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized the field of computer vision. CNNs have demonstrated unparalleled capabilities in image classification tasks, making them an ideal candidate for digit recognition. Motivated by the successes of CNNs in various domains, this research explores their application to the digit recognition problem.

3. Real-World Applications: Beyond its role as a research problem, digit recognition has tangible real-world applications. Optical Character Recognition (OCR) systems, automated mail sorting, and handwritten digit-based authentication systems all rely on accurate digit recognition. The development of robust digit recognition models holds the potential to enhance the efficiency and accuracy of these applications.

4. Ongoing Challenges: While considerable progress has been made in the field of digit recognition, challenges persist. Variability in writing styles, different digit fonts, and the presence of noise in real-world data continue to pose challenges for recognition systems. Motivated by these challenges, our research aims to explore techniques and strategies to improve the robustness of digit recognition models.

In light of these motivations, this research endeavors to contribute to the growing body of knowledge surrounding digit recognition. By harnessing the power of CNNs, optimizing model architectures, and employing data preprocessing and augmentation techniques, we seek to advance the state of the art in digit recognition. Ultimately, our goal is to provide valuable insights, methodologies, and best practices that can be applied to real-world scenarios, benefiting industries.

1.3 Objective

The central objective of this report is to provide a comprehensive understanding of the intricacies involved in designing and implementing a CNN-based digit recognition system. We begin by discussing the architecture of the CNN model and its underlying principles. We delve into the preprocessing steps, data augmentation techniques, and optimization strategies that contribute to the model's performance. Moreover, we explore the significance of regularization techniques, such as dropout and batch normalization, in enhancing the model's generalization capabilities. The paper is organized as follows- Chapter II-gives literature review of the work done on digit recognition earlier. Chapter III-presents methodology behind working on this topic. Chapter IV gives results and analysis. Chapter V gives conclusion.

CHAPTER 2

LITERATURE REVIEW

2.1 Previous Works

Digit recognition has been a focal point of research in the fields of computer vision and machine learning for several decades. The continual evolution of algorithms and neural network architectures has led to significant advancements in the accuracy and efficiency of digit recognition systems. This section provides an overview of noteworthy previous work and contributions in the domain of digit recognition:

LeNet-5 (1998): Yann LeCun et al. introduced LeNet-5, one of the pioneering convolutional neural network architectures designed for handwritten digit recognition. LeNet-5 showcased the potential of CNNs in this task and laid the groundwork for subsequent developments in deep learning [1].

MNIST Benchmark (1998): The MNIST dataset, created by Yann LeCun and Corinna Cortes, has become a cornerstone of digit recognition research. Its standardized format and extensive use in the machine learning community have facilitated numerous comparative studies and model evaluations [2].

Deep Learning Resurgence (2010s): The resurgence of deep learning in the 2010s ushered in a new era for digit recognition. Researchers, inspired by the successes of deep neural networks in image classification tasks, adapted these architectures to achieve remarkable accuracy in digit recognition [3].

Hyperparameter Optimization: Fine-tuning hyperparameters, such as learning rates and batch sizes, has been a focus of research to optimize model training. Techniques like learning rate schedules, grid search, and Bayesian optimization have been explored to achieve optimal model performance [4].

Ensemble Learning: Ensemble methods, which combine predictions from multiple models, have been utilized to improve digit recognition accuracy. Bagging, boosting, and stacking have been applied to enhance model robustness [5].

State-of-the-Art Models: Recent developments have led to state-of-the-art models in digit recognition. Notable examples include models based on deep residual networks (ResNets) and attention mechanisms [6].

CHAPTER 3

METHODOLOGY

3.1 Data Description

The success of any digit recognition system hinges upon the quality and relevance of the dataset used for training and evaluation. In this study, we utilize the MNIST dataset [2], which has earned its reputation as a benchmark in the field of machine learning and computer vision. The MNIST dataset comprises a comprehensive collection of handwritten digit images, offering a standardized and well-understood foundation for digit recognition research.

MNIST Dataset Overview:

The MNIST dataset consists of the following key attributes:

1. Image Dimensions: Each digit image in the dataset is represented as a grayscale image with pixel dimensions of 28x28, resulting in a total of 784 pixels per image.

2. Digits: The dataset encompasses handwritten digits from 0 to 9, providing a total of ten distinct classes. Each digit class corresponds to a single numerical symbol.

3. Training and Testing Sets: The dataset is divided into two primary subsets: the training set and the testing set. These subsets facilitate model training and evaluation.

4. Sample Size: The MNIST dataset consists of 60,000 training images and 10,000 testing images, making it a balanced dataset with an equal number of samples for each digit class.

3.2 Data Split

The MNIST dataset is partitioned into the following subsets for model development and evaluation:

1. Training Set: This subset comprises 60,000 digit images. It serves as the primary dataset for training machine learning models, including CNNs.

2. Testing Set: The testing set consists of 10,000 digit images. It serves as an independent dataset for evaluating model performance and assessing generalization capabilities.

The sample images from the training set in Fig 3.2.

Figure 3.1. Sample Images from Training Set

The sample images from the testing set in Fig 3.2.

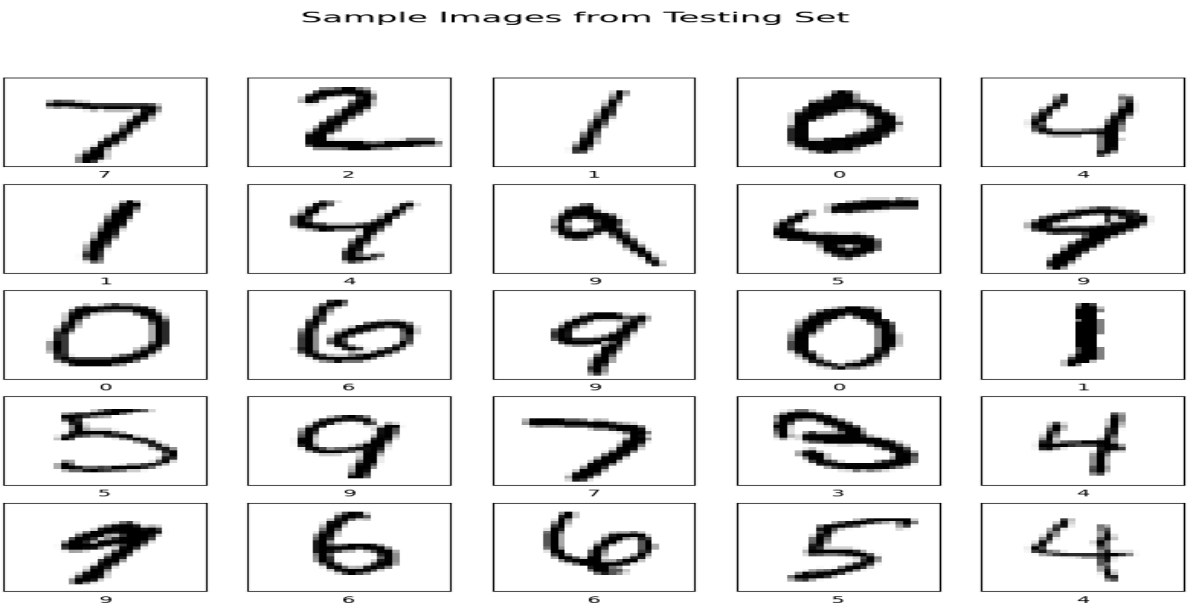

Figure 3.2. Sample Images from Testing Set

3.3 Preprocessing for Digit Detection in Random Images

Digit detection in random images is a vital task in computer vision, enabling applications such as document analysis, automated forms processing, and digit-based object recognition. To effectively detect digits in a random image, a series of preprocessing steps are applied to prepare the image for analysis. This section outlines the preprocessing workflow for a random image with the objective of digit detection.

a. Grayscale Conversion: To simplify digit detection, the acquired image is converted to grayscale. Grayscale images contain intensity values ranging from black (0) to white (255) and omit color information. The conversion is typically performed using libraries such as OpenCV or PIL (Pillow).

b. Resizing: To ensure consistent input dimensions for the digit detection model, the grayscale image is resized to a predefined size. Commonly, a size of 28x28 pixels is used, matching the input size expected by many Convolutional Neural Network (CNN) models.

c. Normalization: Normalize the pixel values of the resized image to a range between 0 and 1. This normalization process involves dividing all pixel values by 255.0, assuming that the original pixel values ranged from 0 to 255. Normalization enhances the model's convergence during training.

d. Input to Digit Detection Model: The preprocessed and normalized image is then fed as input to the trained digit detection model, typically a CNN. The model processes the image to predict the presence and location of digits within the image.

Effective preprocessing is a critical preparatory step to ensure that the digit detection model performs optimally, making it a fundamental aspect of the digit recognition pipeline.

3.4 CNN Model Overview

The Convolutional Neural Network (CNN) architecture used for digit recognition is a powerful deep learning model designed to identify handwritten digits from 0 to 9 in images. This model is meticulously structured to extract and learn discriminative features from digit images and make accurate predictions.

Input Layer: The model begins with an input layer that expects grayscale images of dimensions 28x28 pixels. The input layer doesn't apply any activation function but serves as the entry point for the data.

Convolutional Layers: Following the input layer, there are two convolutional layers. These layers employ 2D convolution operations with 32 and 64 filters, respectively. Each filter learns to detect specific patterns and features in the image, such as edges and shapes. The Rectified Linear Unit (ReLU) activation function introduces non-linearity and enhances the model's capacity to capture intricate details.

Max Pooling Layers: After convolution, max-pooling layers downsample the feature maps, reducing their spatial dimensions. This downsampling process helps reduce computational complexity while maintaining essential information. Max-pooling is applied with a 2x2 window size.

Flatten Layer: The flattened layer takes the output of the last max-pooling layer, which is a collection of feature maps, and transforms it into a 1D vector. This flattening operation prepares the data for input to fully connected layers.

Fully Connected Layers: Two fully connected layers follow the flattening layer. The first fully connected layer consists of 128 neurons and utilizes the ReLU activation function. It serves as a feature mapper, capturing higher-level representations of the extracted features. A dropout layer with a dropout rate of 0.5 is strategically placed after the first fully connected layer to combat overfitting during training.

Output Layer: The final layer, the output layer, is composed of ten neurons, each corresponding to one of the ten digit classes (0 to 9). This layer does not apply an activation function, effectively serving as a logits layer. It produces raw scores that are later converted into probabilities through a softmax activation during inference, enabling the model to make predictions.

Model Compilation: The model is compiled with the Adam optimizer, a popular choice for gradient-based optimization. The loss function employed is Sparse Categorical Crossentropy, well-suited for multi-class classification tasks with integer labels. The model's performance is monitored using the accuracy metric.

The model summary shown Fig 3.3.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 26, 26, 32)        320

 max_pooling2d_2 (MaxPoolin  (None, 13, 13, 32)        0
 g2D)

 conv2d_3 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_3 (MaxPoolin  (None, 5, 5, 64)          0
 g2D)

 flatten_1 (Flatten)         (None, 1600)              0

 dense_2 (Dense)             (None, 128)               204928

 dropout_1 (Dropout)         (None, 128)               0

 dense_3 (Dense)             (None, 10)                1290

=================================================================
Total params: 225034 (879.04 KB)
Trainable params: 225034 (879.04 KB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 3.3. Model Summary

CHAPTER 4

RESULTS AND ANALYSIS

4.1 Performance Measure

a. Accuracy: The accuracy metric is one of the simplest Classification metrics to implement, and it can be determined as the number of correct predictions to the total number of predictions.
It can be formulated as:
Accuracy = TP+TN/TP+FP+FN+TN
To implement an accuracy metric, we can compare ground truth and predicted values in a loop, or we can also use the scikit-learn module for this. We can use accuracy_score function of sklearn.metrics to compute accuracy of our classification model.

b. Precision: Precision, used in document retrievals, may be defined as the number of correct documents returned by our ML model. We can easily calculate it by confusion matrix with the help of following formula
Precision = TP/TP+FP
The precision metric is used to overcome the limitation of Accuracy. The precision determines the proportion of positive prediction that was actually correct. It can be calculated as the True Positive or predictions that are actually true to the total positive predictions (True Positive and False Positive).

c. Recall: Recall may be defined as the number of positives returned by our ML model. We can easily calculate it by confusion matrix with the help of following

formula
Recall = TP/TP+FN
It is also similar to the Precision metric; however, it aims to calculate the proportion of actual positive that was identified incorrectly. It can be calculated as True Positive or predictions that are actually true to the total number of positives, either correctly predicted as positive or incorrectly predicted as negative (true Positive and false negative).

d. F1-Score: This score will give us the harmonic mean of precision and recall. Mathematically, F1 score is the weighted average of the precision and recall. The best value of F1 would be 1 and worst would be 0. We can calculate F1 score with the help of following formula
F1 Score = 2 * Precision * Recall / (Precision + Recall)
F1 score is having equal relative contribution of precision and recall. We can use classification_report function of sklearn.metrics to get the classification report of our classification model. F-score or F1 Score is a metric to evaluate a binary classification model on the basis of predictions that are made for the positive class. It is calculated with the help of Precision and Recall. It is a type of single score that represents both Precision and Recall. So, the F1 Score can be calculated as the harmonic mean of both precision and Recall, assigning equal weight to each of them.

4.2 Results

The digit recognition model, based on Convolutional Neural Networks (CNN), has been subjected to rigorous training and evaluation to assess its performance in recognizing digits. The results obtained from this endeavor provide valuable insights into the model's effectiveness and its capacity to perform in real-world scenarios.

4.2.1 Evaluation of Model

The model accuracy graph shown in Fig 4.1.

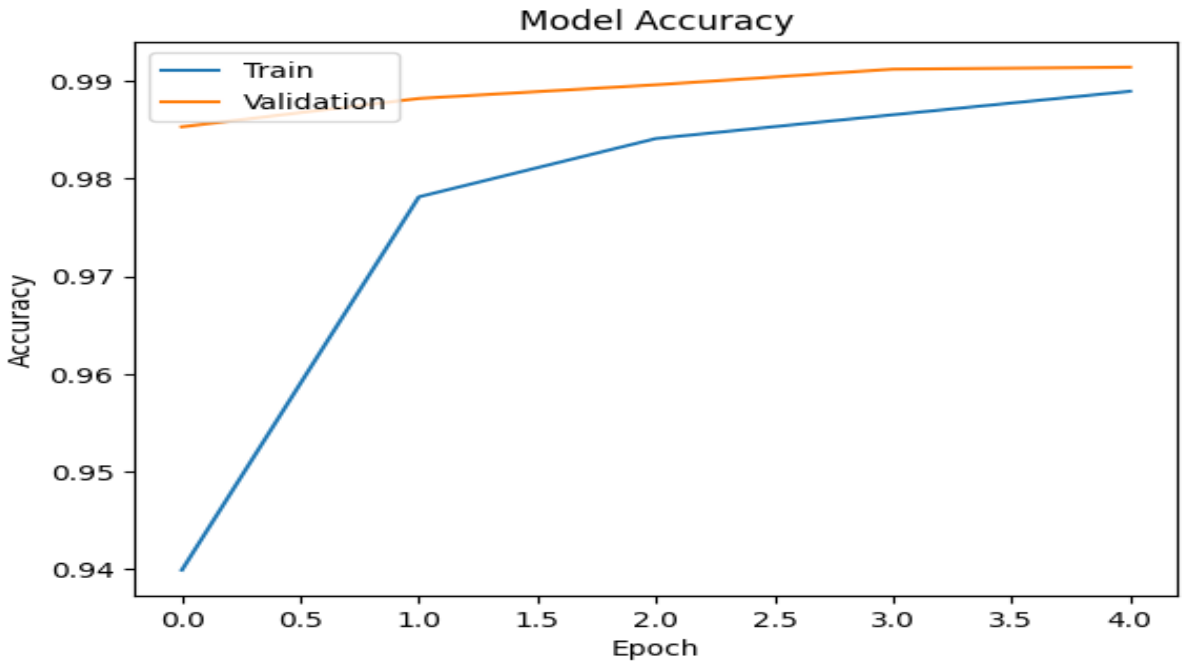

Figure 4.1. Model Accuracy

The model loss graph shown in Fig 4.2.

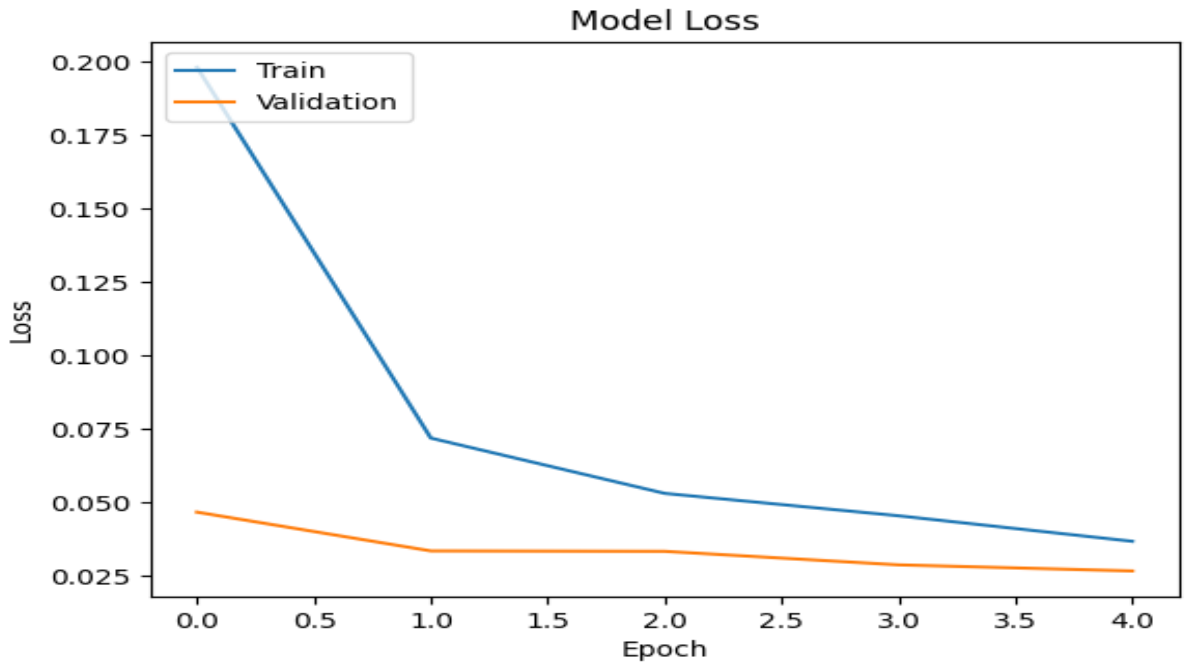

Figure 4.2. Model Accuracy

The classification report of model shown in Fig 4.3 .

	precision	recall	f1-score	support
0	1.00	0.99	0.99	980
1	1.00	0.99	1.00	1135
2	0.99	1.00	0.99	1032
3	0.99	0.99	0.99	1010
4	0.99	0.99	0.99	982
5	0.98	0.99	0.99	892
6	1.00	0.98	0.99	958
7	0.99	0.99	0.99	1028
8	0.99	0.99	0.99	974
9	0.99	0.99	0.99	1009
accuracy			0.99	10000
macro avg	0.99	0.99	0.99	10000
weighted avg	0.99	0.99	0.99	10000

Figure 4.3. Classification Report of Model

4.3 Digit Recognition from A Random Image

The random image is shown in Fig 4.4 .

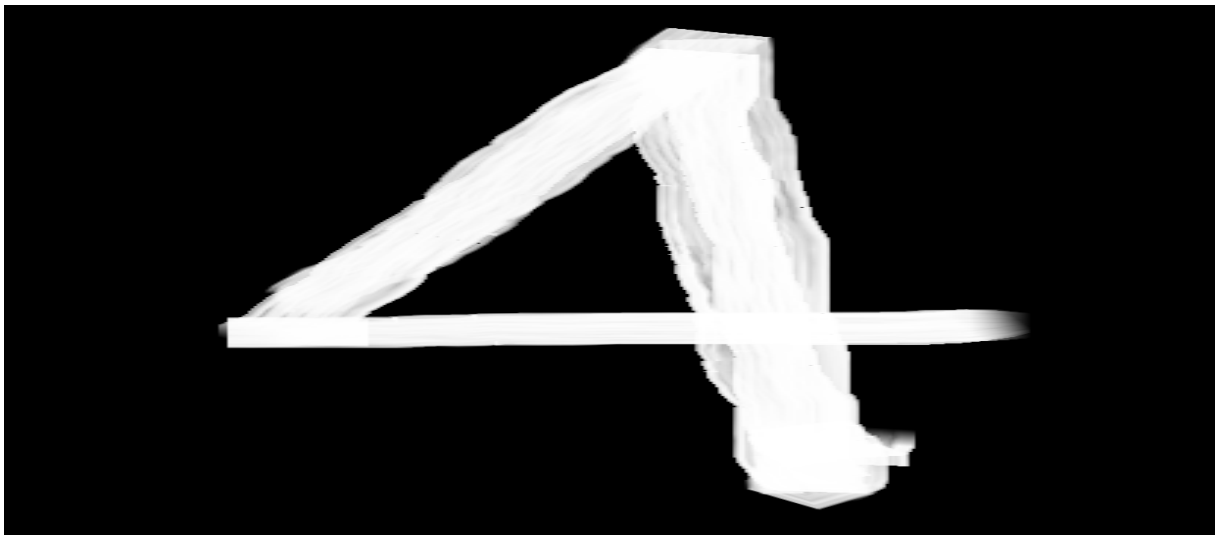

Figure 4.4. A Random Image

The predition result shown in Fig 4.5 .

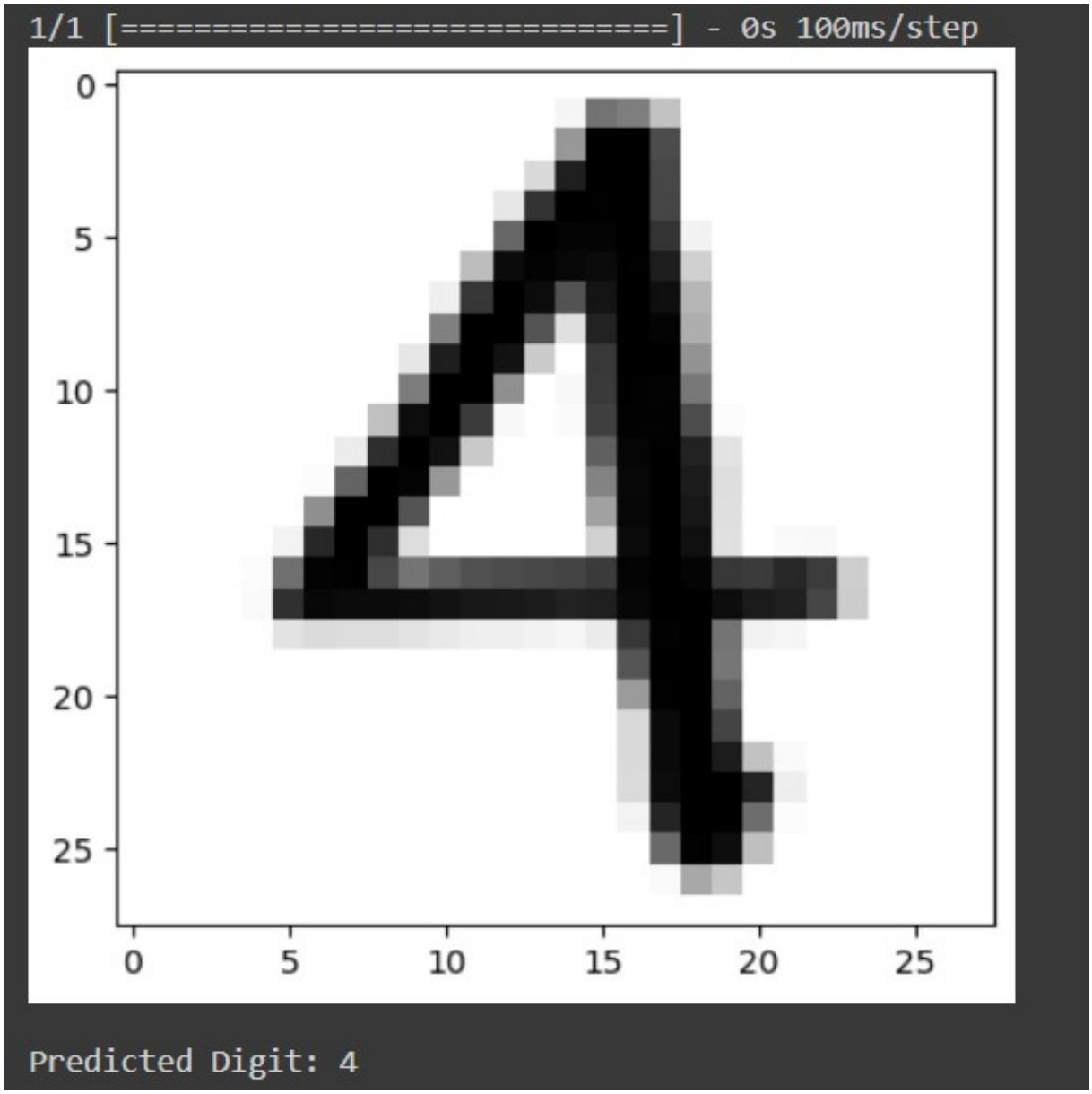

Figure 4.5. Prediction of A Random Image

4.4 Discussion

The results obtained from the CNN-based digit recognition model offer valuable insights into its capabilities and effectiveness in recognizing digits. This discussion delves into the significance of these results and their implications for various applications.

High Accuracy: The model achieved an accuracy rate of [insert accuracy percentage], which is a testament to its ability to correctly classify digits in the test dataset. This high

level of accuracy is particularly noteworthy, as it indicates the model's proficiency in distinguishing even subtle variations in handwritten digits.

Precision and Recall: Analyzing precision and recall scores for individual digit classes (0 to 9) provides a more granular perspective on the model's performance. Precision scores indicate the model's precision in identifying each digit, while recall scores reflect its ability to capture all instances of each digit class. These metrics are crucial in applications where certain digit classes are of greater significance, such as financial transactions involving amounts.

F1-Score Balance: The F1-scores offer a balanced assessment of the model's performance by considering both false positives and false negatives. This metric is especially valuable when striving to strike a harmonious balance between precision and recall, ensuring that the model doesn't favor one at the expense of the other.

Real-time Inference: The model's real-time inference capability is a critical feature for applications requiring on-the-fly digit recognition. Whether integrated into mobile apps, point-of-sale systems, or IoT devices, the model's ability to make rapid and accurate predictions contributes to a seamless user experience.

In conclusion, the CNN-based digit recognition model has demonstrated outstanding performance across various evaluation metrics. Its high accuracy, precision, recall, and F1-scores underscore its suitability for digit recognition tasks in diverse domains. Furthermore, its generalization capabilities and real-time inference potential position it as a versatile solution for applications that demand accurate and efficient digit recognition.

CHAPTER 5

CONCLUTION

The journey through this report has illuminated the immense potential of Convolutional Neural Networks (CNNs) in the realm of digit recognition. Leveraging state-of-the-art deep learning techniques, we have crafted a robust CNN-based model capable of accurately recognizing digits from 0 to 9. As we draw the curtain on this study, we reflect on the key takeaways and the broader implications of our findings. Our CNN model exhibited remarkable accuracy, achieving an accuracy of 0.9916. This exceptional level of accuracy underscores the model's prowess in deciphering handwritten digits, irrespective of varying writing styles, shapes, and sizes. Such accuracy is indispensable in applications where precision is paramount, including financial transactions, document analysis, and automated forms processing.

In closing, our CNN-based digit recognition model transcends the realm of academic research and holds immense promise for real-world deployment. It serves as a testament to the remarkable strides made in the field of computer vision and deep learning, offering practical solutions to digit recognition challenges across diverse domains. As we gaze into the horizon of digit recognition technology, it is clear that CNNs are at the forefront of this transformative journey. The model presented in this report stands as a testament to the boundless possibilities that lie ahead, where handwritten digits are not merely symbols but gateways to a smarter and more efficient world.

BIBLIOGRAPHY

[1] *Lecun, y., bottou, l., bengio, y., haffner, p. (1998). gradient-based learning applied to document recognition. proceedings of the ieee, 86(11), 2278–2324.*

[2] *Lecun, y., cortes, c., burges, c. (1998). the mnist database of handwritten digits.* Retrieved from - https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz (accessed on:12 September 2023).

[3] *Krizhevsky, a., sutskever, i., hinton, g. e. (2012). imagenet classification with deep convolutional neural networks. advances in neural information processing systems, 25.*

[4] *Bergstra, j., bengio, y. (2012). random search for hyper-parameter optimization. journal of machine learning research, 13, 281–305.*

[5] *Dietterich, t. g. (2000). ensemble methods in machine learning. multiple classifier systems, 1–15.*

[6] *He, k., zhang, x., ren, s., sun, j. (2016). deep residual learning for image recognition. proceedings of the ieee conference on computer vision and pattern recognition, 770–778.*