5th Sem Mini Project Report on

HANDWRITTEN CHARACTER RECOGNITION

Submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Student Name: AYUSH GARG **University Roll No.:**

Section: **Class Roll No.:**

Department of Computer Science and Engineering

Graphic Era Hill University

Dehradun, Uttarakhand

2024-25

Problem Statement

Handwritten character recognition is a challenging task in the field of machine learning and computer vision due to the variability in handwriting styles, orientations, and sizes. Traditional methods often fail to deliver high accuracy and robustness, especially for complex and diverse datasets. There is a need for an efficient automated system to accurately recognize handwritten characters, enabling applications such as digitization of handwritten text, automated grading systems, and text-based user interfaces.

Motivation for Doing the Project

1. Widespread Applications: Handwritten character recognition has applications in multiple domains, including postal address reading, banking (e.g., check processing), education (e.g., automated grading), and document digitization. Developing such a system contributes significantly to automating repetitive tasks.
2. Learning Opportunity: This project presents an opportunity to delve into convolutional neural networks (CNNs), a core aspect of deep learning, and their ability to extract meaningful features from image data.
3. Addressing Limitations of Existing Methods: Many traditional algorithms struggle with variations in handwriting styles. By leveraging modern neural networks, this project aims to provide a robust and scalable solution.
4. Practical Implementation: The project bridges theoretical learning with practical application, utilizing Python and deep learning frameworks to address a real-world problem.
5. Personal Growth: Developing this project enhances skills in data preprocessing, neural network design, and model evaluation, essential for a career in AI/ML.

Chapter 1: Introduction

1.1 Scope of the Work

The Handwritten Character Recognition (HCR) project addresses the challenge of digitizing handwritten text. By automating this task, the system enhances efficiency in document processing and eliminates errors introduced during manual transcription.

1.2 Importance

HCR systems have significant applications in postal services, banking, education, and archival work. They provide a way to bridge the gap between handwritten data and digital systems.

1.3 Relation to Previous Work

Earlier approaches to HCR primarily relied on manual feature extraction and statistical models. However, recent advancements in deep learning, specifically Convolutional Neural Networks (CNNs), have revolutionized this field, allowing for automatic feature extraction and improved accuracy.

1.4 Present Developments

This project employs a CNN-based model, trained on the EMNIST Balanced dataset, to accurately recognize handwritten characters. The model demonstrates high performance and is robust across diverse handwriting styles.

Chapter 2: Objectives

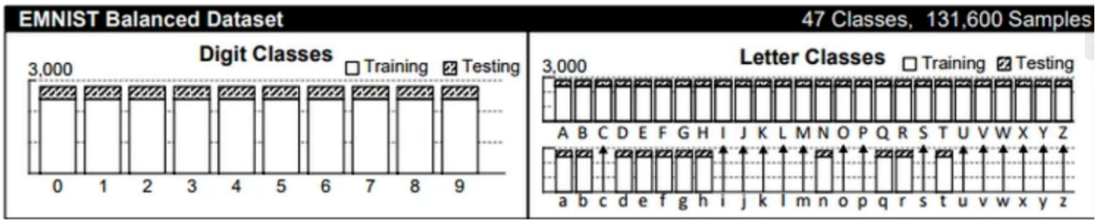
2.1 Project Objectives

1. To develop a robust machine learning model for handwritten character recognition.
2. To achieve an accuracy exceeding 95% using the EMNIST Balanced dataset.
3. To evaluate the model's performance on unseen data and demonstrate real-world applicability.

Chapter 3: Methodology

3.1 Dataset Description

The EMNIST Balanced dataset comprises 47 classes, including uppercase and lowercase letters and digits. It contains 131,600 samples distributed uniformly to address class imbalance.



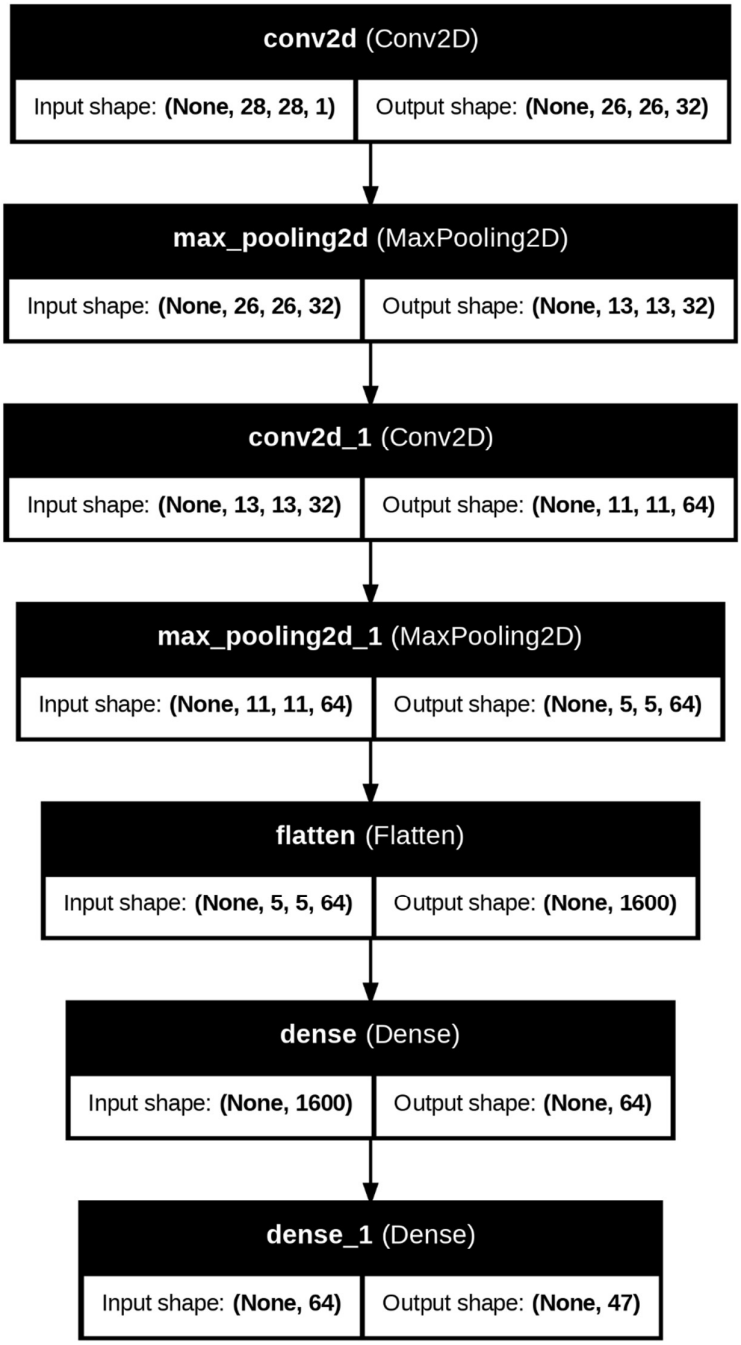
3.2 Preprocessing

1. **Normalization**: Pixel values were scaled to the range $[0, 1]$.
2. **Reshaping**: Images were resized to 28×28 pixels.
3. **Encoding**: Labels were converted into a one-hot encoded format.

3.3 Model Architecture

The Convolutional Neural Network (CNN) consists of:

- Input layer: Accepts 28×28 grayscale images.
- Convolutional layers: Extract features using 32 and 64 filters.
- Max-pooling layers: Down sample feature maps.
- Dropout layer: Reduces overfitting.
- Fully connected layers: Map features to classes.
- Output layer: Utilizes SoftMax activation for classification.



3.4 Tools Used

- **Programming Language**: Python
- **Libraries**: TensorFlow/Keras, NumPy, Matplotlib
- **Development Environment**: Jupyter Notebook

Chapter 4: Implementation

4.1 Training Parameters

1. **Loss Function**: Categorical Cross-Entropy
2. **Optimizer**: Adam ($learning\ rate\ =\ 0.001$)
3. **Batch Size**: 128
4. **Epochs**: 20
5. **Validation Split**: 20%

4.2 Process

The model was trained for 30 epochs, monitoring accuracy and loss. Early stopping was used to prevent overfitting.

4.3 Equations

The CNN optimizes the following loss function:

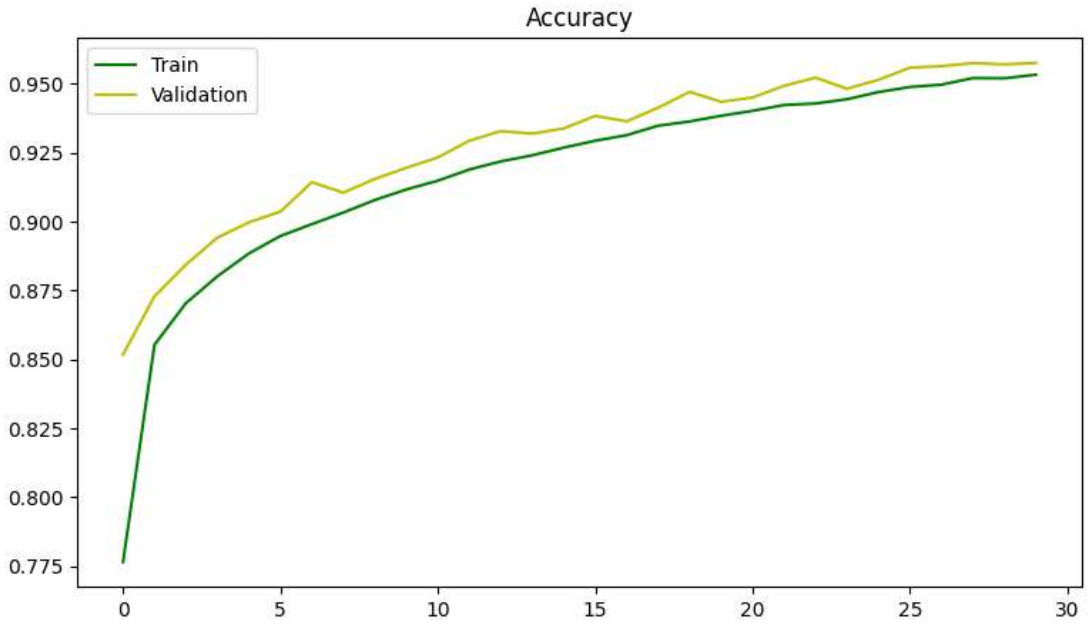
$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{4.1}$$

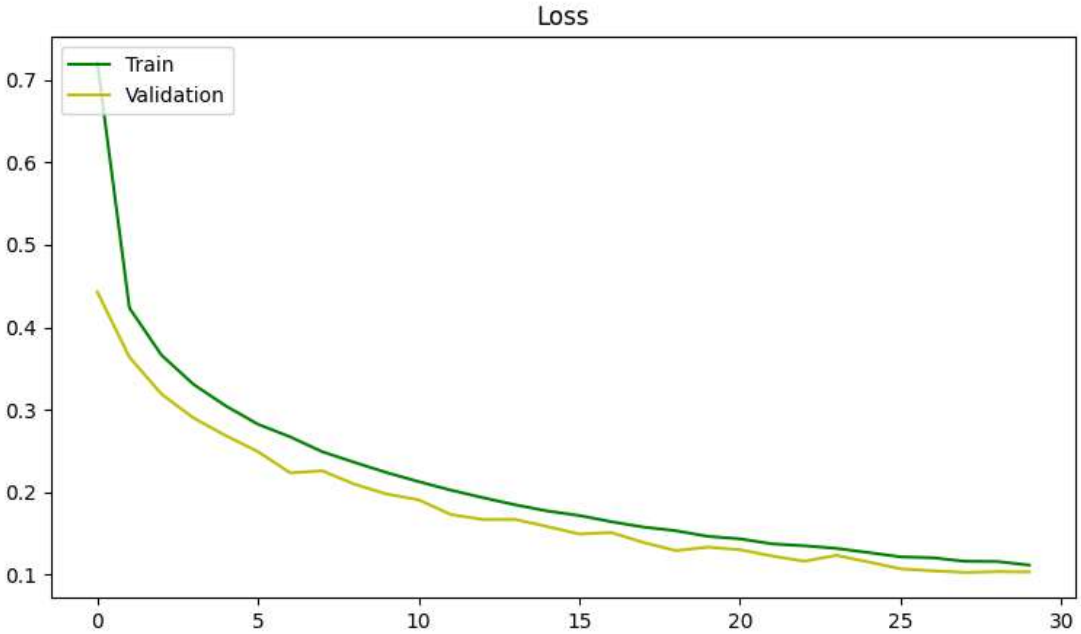
Where y_i is the true label and $\hat{y}_i$ is the predicted probability.

Chapter 5: Results

5.1 Performance Metrics

1. **Training Accuracy**: 95.33%
2. **Validation Accuracy**: 95.76%
3. **Test Accuracy**: 95.76%





5.2 Sample Outputs

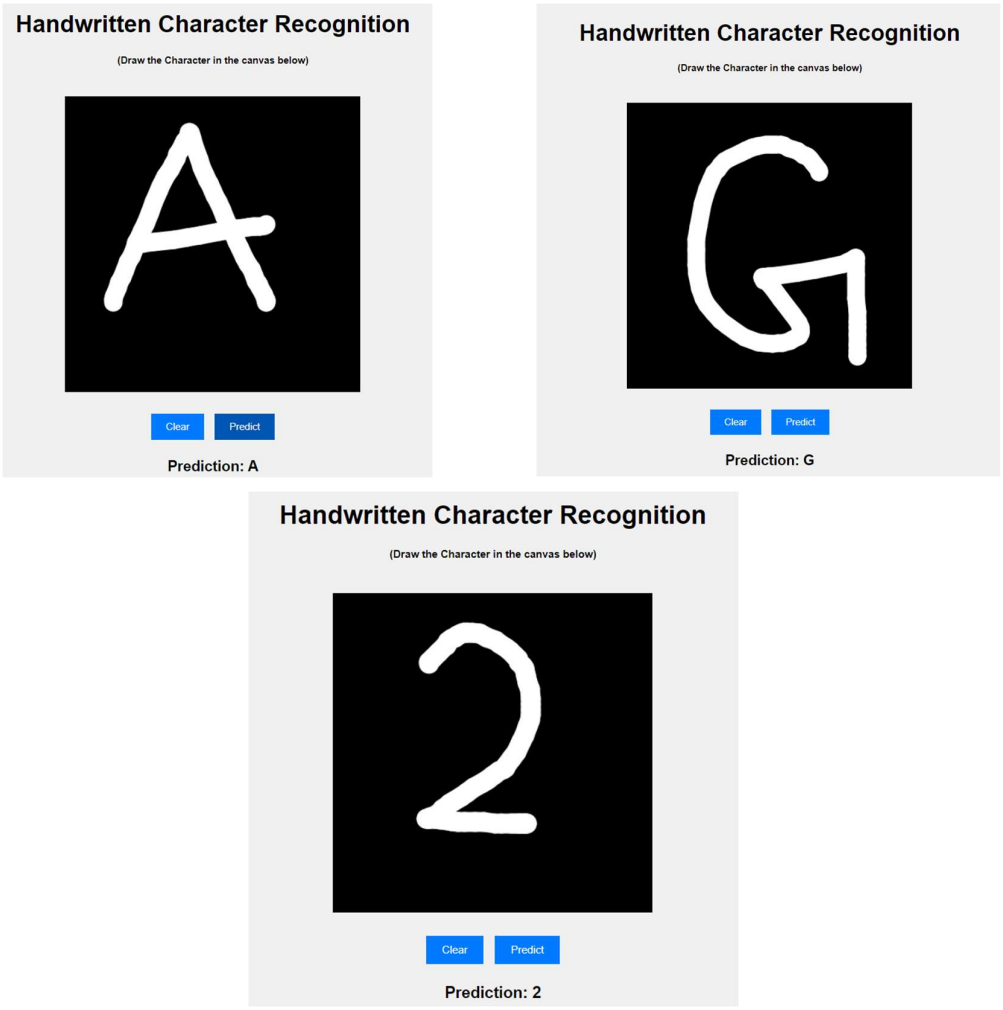


Figure 5.1 illustrates the predicted vs. actual characters.

Chapter 6: Challenges

6.1 Issues Faced

1. **Class Imbalance**: Addressed by selecting the EMNIST Balanced dataset.
2. **Overfitting**: Mitigated using dropout layers and early stopping.
3. **Handwriting Variability**: Resolved by training on a diverse dataset.

Chapter 7: Applications

1. Optical Character Recognition (OCR) for handwritten text digitization.
2. Automated data entry for banking and educational institutions.
3. Assisting disabled individuals by converting handwriting to text.

Chapter 8: Conclusion and Future Work

8.1 Conclusion

The CNN-based HCR system achieves high accuracy of 95.76% and demonstrates its utility in recognizing handwritten characters.

8.2 Future Enhancements

1. Extend the dataset to include more diverse handwriting styles.
2. Develop a user-friendly interface for real-world deployment.
3. Explore transfer learning for faster model training.

References

1. Cohen, G., et al. "EMNIST: An extension of MNIST to handwritten letters." arXiv preprint arXiv:1702.05373 (2017).
2. TensorFlow Documentation.