Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 5 | May-2026
ISSN (O) 3107-6696

Real-Time Handwritten Character Recognition Using Convolutional Neural Networks

Dr. Archana Kumar, Bhavnay Gupta
profdrarchanakumar@gmail.com, bhavnaygupta12382@gmail.com
Supervisor & Head of Department
Department of Artificial Intelligence and Data Science,
Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi

---

Abstract - This paper presents a real-time handwritten character recognition system capable of identifying handwritten English alphabets (A-Z) and numeric digits (0-9) using deep learning. The system employs two separate Convolutional Neural Network (CNN) models — one trained on the A-Z Handwritten Character Dataset comprising 372,450 samples and another trained on the MNIST dataset comprising 60,000 digit samples. A confidence-based router mechanism determines whether a drawn character is a letter or digit by comparing output probabilities of both models and selecting the prediction with higher confidence. Both models achieve 99% validation accuracy after 5 training epochs. A real-time graphical user interface built using Tkinter allows users to draw characters on a digital canvas and receive instant predictions. Key contributions include an auto-crop preprocessing technique that eliminates scale sensitivity, a dual-model confidence router for letter-digit disambiguation, and contour-based character segmentation for word and sentence level recognition.

I. INTRODUCTION

Handwritten character recognition (HCR) has been a fundamental challenge in the fields of computer vision and pattern recognition for several decades. The ability to automatically interpret and digitize handwritten text has widespread applications in document digitization, postal automation, healthcare record management, banking, and accessibility tools for differently abled individuals.

With the rapid advancement of deep learning, particularly Convolutional Neural Networks (CNNs), it has become possible to build high-accuracy character recognition systems that learn directly from raw pixel data without the need for manual feature engineering. CNNs have consistently outperformed traditional machine learning approaches such as Support Vector Machines and template matching on standard handwriting recognition benchmarks.

This paper presents a real-time handwritten character recognition system that combines two specialized CNN models through a confidence-based routing mechanism. The system supports recognition of all 26 uppercase English alphabets and 10 numeric digits through an interactive digital drawing canvas and extends to word and sentence level recognition through contour-based character segmentation.

A. Challenges

- Handwritten characters vary significantly between individuals in terms of size, stroke thickness, style, and orientation, making generalization difficult.
- Certain letters and digits are visually similar — O and 0, I and 1, S and 5 — creating disambiguation challenges for a single combined model.
- The gap between training data format and live canvas input format causes preprocessing mismatches that degrade real-time accuracy.
- Combined datasets such as EMNIST ByClass have label mapping inconsistencies in CSV format that cause systematic misclassification.
- Scale sensitivity — small drawings on a large canvas become tiny after resizing — reduces prediction accuracy without appropriate preprocessing.

B. Need for This System

The increasing digitization of education, healthcare, and business processes has created a growing need for reliable handwriting recognition tools. Traditional approaches either require manual feature engineering or suffer from dataset-specific issues when attempting to combine letter and digit recognition into a single model. There is a clear requirement for a clean, modular, and real-time system that reliably recognizes both handwritten alphabets and digits through an interactive interface, with the ability to build words and sentences progressively.

II. LITERATURE REVIEW

The evolution of handwritten character recognition has progressed from simple template matching to advanced deep learning architectures. Early systems relied on rule-based approaches and statistical methods that required manual feature engineering and struggled to generalize across different writing styles.

LeCun et al. introduced LeNet-5, one of the earliest CNN architectures designed for handwritten digit recognition. Trained on the MNIST dataset, it demonstrated that convolutional layers could learn spatial features directly from

raw pixel data, achieving significantly better accuracy than fully connected networks and establishing the foundation for all subsequent CNN-based character recognition research.

Cohen et al. introduced the EMNIST dataset as an extension of MNIST to include handwritten letters in addition to digits. The ByClass split contains 814,255 samples across 62 classes. While widely adopted in research, the CSV format of EMNIST ByClass has label mapping inconsistencies that require careful handling, as integer class labels do not follow a straightforward sequential mapping.

Baldominos et al. compared various CNN architectures on EMNIST and found that models with three or more convolutional layers consistently outperformed shallower architectures, achieving a best accuracy of 86.75% on the ByClass split. The proposed dual-model approach achieves 99% accuracy on both the A-Z and digit datasets independently by avoiding the label mapping issues inherent in the combined EMNIST format.

Chakraborty et al. proposed a real-time recognition system using OpenCV for character capture and a CNN for classification, achieving 85% accuracy on live input but suffering from sensitivity to scale variation. The auto-crop preprocessing introduced in this work directly addresses this limitation.

III. OBJECTIVES AND SCOPE OF WORK

A. Objectives

The central objective of this project is to design, develop, and evaluate a real-time handwritten character recognition system that reliably identifies handwritten English alphabets (A-Z) and numeric digits (0-9) through an interactive drawing canvas. Specific objectives are:

- To train a CNN model on the A-Z Handwritten Character Dataset for uppercase letter recognition with a minimum validation accuracy of 95%.
- To train a separate CNN model on MNIST for digit recognition with a minimum validation accuracy of 95%.
- To implement a confidence-based router that intelligently selects between both models for each input character.
- To develop an auto-crop preprocessing pipeline that eliminates sensitivity to drawing scale and position.
- To build a real-time GUI with single character, word, and sentence recognition modes.

B. Scope of Work

The scope of this project covers the complete development of the recognition system from dataset preparation to real-time deployment. The system focuses on uppercase English alphabets and numeric digits, supporting single character prediction, word building through contour-based segmentation, and sentence construction through a progressive space-separated word accumulation interface. Lowercase letter recognition, cursive handwriting, and web-based deployment are identified as future work.

IV. METHODOLOGY

The overall development workflow is divided into the following major components: Dataset Selection and Preparation, Data Preprocessing, CNN Model Design, Model Training, Confidence-Based Router, Auto-Crop Preprocessing, Character Segmentation, and GUI Development.

A. Dataset Selection

Two publicly available datasets were selected. The A-Z Handwritten Character Dataset contains 372,450 samples of handwritten uppercase alphabets across 26 classes stored as a CSV file with each row representing a flattened 28x28 grayscale image. The MNIST dataset contains 70,000 handwritten digit images across 10 classes, available directly through the Keras API.

The EMNIST ByClass dataset was initially evaluated but was found to have label mapping inconsistencies in its CSV format causing consistent misclassification. Two separate verified datasets with clean label mappings were used instead.

Dataset	Classes	Samples	Accuracy
A-Z Handwritten	26 (A-Z)	372,450	99%
MNIST Digits	10 (0-9)	70,000	99%

TABLE I: Dataset Summary

B. Data Preprocessing

Both datasets were preprocessed through: reshaping pixel arrays from (N, 784) to (N, 28, 28, 1); normalizing pixel values by dividing by 255.0; one-hot encoding labels using Keras to_categorical; and splitting the A-Z dataset 80/20 using train_test_split. MNIST uses its pre-defined Keras split.

C. CNN Architecture

Both models share the same CNN architecture differing only in output neurons. Three convolutional blocks extract progressively complex features — edges in block 1, stroke combinations in block 2, and complete character shapes in block 3. MaxPooling after each block reduces spatial dimensions and provides translation invariance. Two fully connected Dense layers combine all features for the final Softmax classification.

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 5 | May-2026
ISSN (O) 3107-6696

Layer	Type	Config
1	Conv2D	32 filters, 3x3, ReLU
2	MaxPool2D	2x2, stride 2
3	Conv2D	64 filters, 3x3, ReLU
4	MaxPool2D	2x2, stride 2
5	Conv2D	128 filters, 3x3, ReLU
6	MaxPool2D	2x2, stride 2
7	Flatten	3D to 1D
8	Dense	64 neurons, ReLU
9	Dense	128 neurons, ReLU
10	Output	26/10 neurons, Softmax

TABLE II: CNN Architecture

D. Training Configuration

Both models were compiled with the Adam optimizer (learning rate = 0.001) and categorical cross-entropy loss. ReduceLROnPlateau reduces the learning rate by a factor of 0.2 when validation loss plateaus for 2 epochs. EarlyStopping terminates training when validation loss does not improve for 4 epochs and restores best weights. Both models converged within 5 epochs with batch size 128.

E. Confidence-Based Router

The router runs both models on every input simultaneously. It compares the peak confidence of each model's output probability distribution and selects the prediction from whichever model is more confident. This exploits the specialization of each model — the letter model is highly confident on letter inputs and the digit model on digit inputs — enabling reliable disambiguation of visually similar pairs such as O/0 and I/1.

F. Auto-Crop Preprocessing

Characters drawn on the 400x400 canvas are auto-cropped by detecting the bounding box of all dark pixels, tightly cropping to that region, adding 20 pixels of uniform padding, and resizing to 28x28 using LANCZOS resampling. This ensures consistent character scale regardless of drawing size or position, matching the training data distribution.

G. Character Segmentation

In word and sentence mode, OpenCV contour detection segments the canvas into individual character regions sorted left to right. Each contour bounding box above a minimum size threshold is cropped, padded, and processed through the full preprocessing and prediction pipeline independently. Predicted characters are concatenated to form the recognised word.

V. RESULTS AND DISCUSSION

A. Model Performance

Model	Val Acc	Loss	Epochs
Letter CNN (A-Z)	99.0%	0.031	5
Digit CNN (MNIST)	99.0%	0.018	5

TABLE III: Performance Results

Both models achieved 99% validation accuracy after 5 training epochs, significantly outperforming the EMNIST-based baseline of 86.75% reported by Baldominos et al. The letter model's accuracy improvement is attributed to the cleaner label mapping and the absence of confusing lowercase-uppercase mixed classes in the A-Z dataset.

B. Comparison with Existing Approaches

Method	Dataset	Accuracy
Template Matching	Custom	~60%
SVM + HOG	MNIST	~98%
LeNet-5	MNIST	99.05%
CNN + EMNIST	EMNIST ByClass	86.75%
Proposed (Letters)	A-Z Dataset	99.0%
Proposed (Digits)	MNIST	99.0%

TABLE IV: Comparison with Existing Methods

C. Challenges and Solutions

Challenge	Solution
EMNIST label mismatch	Two separate verified

	datasets
Preprocessing mismatch	Systematic inversion testing
Scale sensitivity	Auto-crop before resize
Letter-digit confusion	Confidence-based dual router

TABLE V: Challenges and Solutions

VI. TENTATIVE CHAPTERIZATION

Chapter 1 — Introduction: Background, motivation, problem statement, objectives, and scope of the project. Defines the significance of real-time handwriting recognition.

Chapter 2 — Literature Review: Reviews existing research in handwritten character recognition, CNN architectures, MNIST and EMNIST benchmarks, and real-time recognition systems.

Chapter 3 — System Analysis: Software requirements specification covering functional and non-functional requirements, feasibility study, and dataset selection rationale.

Chapter 4 — System Design: Full system architecture including the preprocessing pipeline, dual CNN model design, confidence-based router, character segmentation module, and GUI design.

Chapter 5 — Implementation: Tools and technologies used, module descriptions, key code snippets for the preprocessing pipeline, router, segmentation, and GUI.

Chapter 6 — Results and Testing: Training and validation accuracy graphs, output screenshots, test cases, and performance analysis comparing the proposed system with existing approaches.

Chapter 7 — Conclusion and Future Scope: Summary of findings, system limitations, and planned future enhancements including spell checking, lowercase recognition, and web deployment.

VII. CONCLUSION

This paper presented a real-time handwritten character recognition system using two specialized CNN models connected through a confidence-based router. The system achieves 99% validation accuracy on both the A-Z letter dataset and the MNIST digit dataset after only 5 training epochs, outperforming existing EMNIST-based combined model approaches.

Key innovations include the auto-crop preprocessing technique that eliminates scale sensitivity in live canvas input, the dual-model confidence router for reliable letter-digit disambiguation, and contour-based segmentation for word and sentence recognition. The iterative debugging methodology developed to resolve EMNIST label mapping and preprocessing mismatch issues provides a reusable framework for diagnosing similar problems in any image classification deployment.

Future work will explore spell-checking integration, web deployment using Flask, lowercase letter recognition, and extension to mathematical symbol recognition.

REFERENCES

- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.
- G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: Extending MNIST to Handwritten Letters," IJCNN, 2017.
- A. Baldominos, Y. Saez, and P. Isasi, "A Survey of Handwritten Character Recognition with MNIST and EMNIST," Applied Sciences, vol. 9, no. 15, 2019.
- Sachin Patel, "A-Z Handwritten Alphabets in .csv Format," Kaggle Dataset, 2018. [Online]. Available: https://www.kaggle.com/datasets/sachinpatel21/az-handwritten-alphabets-in-csv-format
- Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," IEEE TPAMI, vol. 35, no. 8, pp. 1798-1828, 2013.
- "Keras Documentation," Keras, 2025, https://keras.io.
- "TensorFlow Documentation," Google, 2025, https://www.tensorflow.org.
- "OpenCV Documentation," OpenCV, 2025, https://docs.opencv.org.