International Journal of Research in Engineering and Innovation (IJREI)

Journal home page: https://ijrei.com/

RESEARCH PAPER

Enhancing OCR Accuracy through Convolutional Learning Using Convolutional Neural Network (CNN)

Md Maskur Alam, Ragib Ahmad, Ismail Javed, Mohd. Shahber Chauhan

Department of Computer Science and Information Technology, Meerut Institute of Engineering and Technology, Meerut, India

Article Information

Received: 09 April 2026
Revised: 01 May 2026
Accepted: 04 May 2026
Available online: 04 May 2026

Keywords:

Optical Character Recognition
Robotic Process Automation
Convolutional Neural Networks
Image Acquisition
RPA-Based Automation

Abstract

Optical Character Recognition (OCR) is a key element in the digital conversion of hand written documents which at the same time presents great difficulty in terms of achieving high accuracy of recognition which we put down to the fact that hand writing styles vary as do character forms and we have image noise. We present in this paper a improved hand written OCR framework which is based on Deep Convolutional Neural Networks (CNNs) for better character recognition. We have designed a model which out of the box -- puts together hierarchical and diagnostic features from gray scale hand written character images via use of many convolutional layers, ReLU activation functions, max-pooling and drop out regularization. We train and test the system on the EMNIST reference set as well as a we created hand written data set which we did to have a standard evaluation process and also to look at real world application. We look at performance through the lenses of accuracy, precision, recall, F1 score and also use confusion matrix analysis. We report high accuracy in recognition, good generalization performance and also see a reduction in overfitting which in turn proves the value of CNN based learning for hand written OCR tasks. Keywords: Optical Character Recognition (OCR), Convolutional Neural Network (CNN), Handwritten Character Recognition, EMNIST Dataset, and Deep Learning

1. Introduction

Optical Character Recognition (OCR) has emerged as a foundational technology in the digital transformation era, enabling the conversion of printed and handwritten textual information into structured, machine-readable formats. This capability has facilitated a wide range of applications, including document digitization, automated form processing in banking systems, archival management, and evaluation processes in educational institutions. Despite the considerable progress achieved in recognizing printed text with high accuracy, handwritten character recognition continues to pose significant challenges. These challenges arise due to variations in writing styles, stroke thickness, orientation, overlapping characters, and the presence of noise or distortions in input images. Consequently, the development of robust and adaptable OCR systems for handwritten text remains an active area of research. Early OCR systems primarily relied on handcrafted feature extraction techniques combined with classical machine learning classifiers such as k-Nearest Neighbors, Support Vector Machines, and Hidden Markov Models. These approaches required manual design of features, including edges, contours, and geometric patterns, which limited their ability to generalize across diverse handwriting styles. While such methods performed adequately for structured and printed text, their effectiveness significantly declined when applied to handwritten data due to its inherent variability and complexity. The advent of deep learning has fundamentally transformed the field of OCR, particularly with the introduction of Convolutional Neural Networks (CNNs). CNNs have demonstrated remarkable capability in learning hierarchical representations directly from raw pixel inputs,

Corresponding author: Md. Maskur Alam
Email Address: abhishek.soam.csit.2022@miet.ac.in
https://doi.org/10.36037/IJREI.2026.10205

eliminating the need for manual feature engineering. Through successive convolutional and pooling layers, CNNs can automatically extract low-level features such as edges and curves, as well as high-level semantic patterns corresponding to characters and symbols. This hierarchical learning process enables CNN-based models to achieve superior performance in recognizing complex handwritten text, even in the presence of noise and distortions. A growing body of literature highlights the effectiveness of CNN-based approaches in OCR applications. Lekshmy et al. [1] developed a CNN-based handwritten character recognition system aimed at improving automated evaluation processes, reporting significant improvements in accuracy compared to traditional methods. Similarly, Nawaz et al. [2] proposed an optimized CNN architecture that emphasizes the importance of network depth and kernel configuration, achieving enhanced recognition performance while reducing computational complexity. Namysl and Konya [3] introduced a dictionary-free deep learning OCR framework, demonstrating strong generalization capabilities for recognizing previously unseen characters and patterns, which is particularly valuable in real-world applications where predefined dictionaries may not be sufficient.

Further advancements have been reported in the integration of deep learning techniques for large-scale and hybrid OCR systems. Li [4] presented an OCR system specifically designed for English text recognition, illustrating how deep neural networks can be effectively deployed in large-scale environments. Kulkarni et al. [5] explored a hybrid approach combining CNNs, Recurrent Neural Networks, and Non-negative Matrix Factorization to enhance the digitization of handwritten notes. Their findings indicate that CNNs are highly effective in extracting spatial features, while sequential models such as RNNs improve the recognition of continuous text sequences. Additionally, Rakuka et al. [6] proposed a framework that integrates CNN-based feature extraction with Transformer-based models, highlighting the complementary role of attention mechanisms in handling sequence-level recognition tasks.

Recent studies further reinforce the superiority of deep learning-based OCR systems. Devi Sri et al. [7] demonstrated that careful tuning of convolutional layers and activation functions can lead to significant improvements in recognition accuracy. Vidhya et al. [8] applied OCR and deep learning techniques to bank cheque signature verification, showing that CNNs are capable of recognizing fine-grained patterns even in noisy environments. Singh et al. [9] utilized a pre-trained VGG19 model and demonstrated that transfer learning significantly enhances OCR performance by leveraging knowledge from large-scale image datasets. Furthermore, S. N et al. [10] emphasized the importance of standardized datasets such as EMNIST for benchmarking and evaluating handwritten character recognition models.

1.1 Data Set Description

Two datasets are utilized to evaluate the performance of the proposed model, ensuring both benchmark validation and real-world applicability. The first dataset, EMNIST (Extended Modified National Institute of Standards and Technology), is widely recognized for handwritten character recognition tasks and consists of a large collection of grayscale images representing digits and letters across multiple classes. Its diversity in writing styles and structured format makes it highly suitable for training and evaluating deep Convolutional Neural Network models. In addition to this benchmark dataset, a custom handwritten dataset is employed to assess the model's practical performance under real-world conditions. This dataset comprises handwritten samples collected from multiple individuals, capturing variations in writing styles, stroke patterns, and character formations. The inclusion of both EMNIST and custom datasets ensures a comprehensive evaluation, demonstrating the model's ability to achieve high accuracy while maintaining strong generalization across diverse and unstructured input data.

2. Problem Statement

Hand in hand with the issue of variability in how people write which includes in stroke thickness, character appearance, and placement of characters we see also that noise, low quality images, and non-uniform illumination play a role in it. These issues in turn cause us to see a drop-in recognition accuracy which is a very large issue when we put our models into devices which have limited resources for computation and memory. In the past we have had traditional OCR and shallow machine learning models do poor job in such settings. This research is put forth to develop an efficient Convolutional Neural Network (CNN) based OCR system which we see will deliver high accuracy, robustness to handwriting variations and noise, and low computational complexity which in turn will be suitable for low end hardware.

3. Methodology

Hand written image capture, image preprocessing, feature extraction which is done via a Convolutional Neural Network (CNN), character classification which is performed by the SoftMax function, and results presentation with visualization.

3.1 CNN Architecture

The architecture diagram which we present shows the step by step process of the put forth Convolutional Neural Network (CNN) we have. We see that the input image goes through many convolutional layers which have learnable filters that at first stage identify basic elements like edges and strokes. As the input moves deeper into the network at each stage of convolutions and pooling we see it learn more complex and abstract character elements. At the end the extracted features are fed into fully connected layers which in turn produce a SoftMax output of the probabilities for each character class. This which we see in the architecture diagram is the progressive transformation of raw pixels to discriminative character features. Fig.1 Architecture of the proposed CNN-based OCR system. The network consists of multiple convolutional layers followed by batch normalization and max-pooling operations for hierarchical feature extraction.

Fully connected layers are used for classification, and the final SoftMax layer outputs class probabilities for handwritten characters.

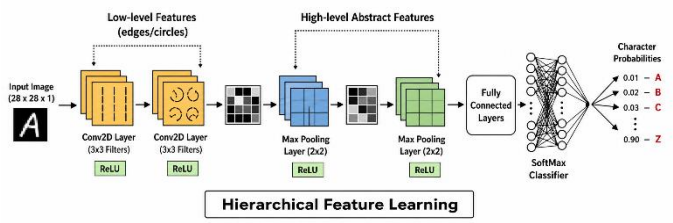

Figure 1: Hierarchical Feature Learning in Convolutional Neural Network for Handwritten Character Recognition

3.2 Data Preprocessing

Data preparation is a base element in the development of a successful CNN based Handwritten Optical Character Recognition (OCR) system. In our study we organized the raw data into separate folders for each character class which in turn facilitated clear labeling and smooth integration with the CNN input pipeline. We also converted each image to gray scale which in that process reduced computational load while at the same time we preserved the main features of the characters and we also resized them to a 28x28 pixel which in that we achieved uniform input for the network. Also we scaled the pixel values to between 0 and 1 which in turn improved the rate of convergence and we also saw to it that gradient propagation during training was stabilized. Also we used one hot encoding for character labels which in that we enabled the network to do multi class classification very well. In order to enhance model robustness to variation in handwriting styles, more diversity in the training data was achieved through data augmentation methods, such as random rotation, translation, and scaling Training, validation and testing sets were used. In the training set we optimized network parameters, in the validation set we tracked model performance during training to avoid over fitting, and in the test set we assessed model results for the final accuracy. We implemented this systematic preprocessing which guarantees that the input data for the CNN is clean, of high quality, standardized and supports accurate and consistent recognition of handwritten characters.

3.3 Data Augmentation

Training, validation and testing sets were used. In the training set we improved network parameters, in the validation set we monitored model performance during training to prevent over fitting, and in the test set we evaluated model results for the final accuracy. We put in this systematic preprocessing which reports that the input data for the CNN is clean, of high quality, standardized and supports accurate and consistent recognition of handwritten characters. Input Handwritten Image Acquisition

The system which has the ability to take in hand written character images from scanned in documents, out of a camera's capture, or real time from a webcam. Also, these images may be of different sizes, rotations, and background which is present. For best results with our model all input images are transformed into the same digital format prior to which they go on for more processing.

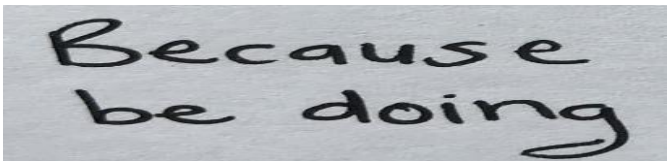

Figure 2: Sample Handwritten Text Image for OCR Recognition

Fig. 2 illustrates the image preprocessing stage, which is essential for enhancing image quality and minimizing unwanted variations. Initially, the acquired images are converted into grayscale to eliminate color information and reduce computational complexity. Subsequently, all images are resized to a uniform resolution of 28 × 28 pixels, ensuring compatibility with the input requirements of the Convolutional Neural Network model. In addition, pixel intensity values are normalized to a range between 0 and 1, which improves training stability and accelerates convergence during model learning. Feature extraction is performed using a Convolutional Neural Network, which automatically learns and represents spatial features from the input images. The initial convolutional layers capture fundamental patterns such as edges and curves, while deeper layers extract more complex structural characteristics of handwritten characters. Max pooling layers are incorporated to reduce the dimensionality of feature maps while preserving significant information, resulting in robust and efficient feature representations.

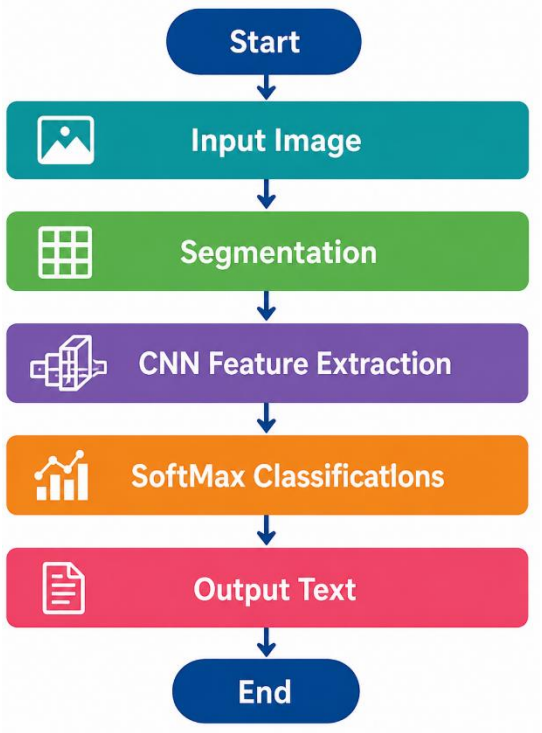

Figure 3: Workflow of CNN-Based Optical Character Recognition System

Fig. 3 illustrates the workflow of a CNN-based Optical Character Recognition system, showing the sequential stages involved in converting an input image into readable text. The process begins with the input image, which typically contains handwritten or printed characters. This image then undergoes segmentation, where individual characters or regions of interest are separated to facilitate accurate processing. Following segmentation, the CNN feature extraction stage is applied, where the model automatically learns and extracts important spatial features such as edges, curves, and complex patterns from the input data. These extracted features are then passed to the SoftMax classification layer, which computes the probability of each character class and identifies the most likely output. Finally, the recognized characters are converted into output text, completing the recognition process. The workflow concludes with the end stage, representing the successful transformation of visual input into structured textual information.

For character classification, the extracted feature maps are flattened and passed through fully connected layers. A SoftMax activation function is applied at the output layer to compute the probability distribution across all character classes, and the class with the highest probability is selected as the final predicted output.

The output prediction is then visualized by mapping the predicted class to its corresponding alphanumeric character. To evaluate model performance, visualization techniques such as training and validation accuracy curves, loss curves, and confusion matrices are utilized. These tools provide valuable insights into the learning behavior and classification performance of the proposed OCR system.

3.4 CNN Layer Architecture

The proposed CNN model consists of multiple convolutional layers followed by pooling layers and fully connected layers. Each convolutional layer applies a set of learnable filters to extract spatial features, while ReLU activation introduces non-linearity. Max-pooling layers reduce spatial dimensions and computational complexity. The high-level features obtained from convolutional blocks are flattened and passed to dense layers for classification.

Fig. 4 illustrates the architecture of a Convolutional Neural Network designed for handwritten character recognition. The process begins with the input layer, where grayscale images of size 28 × 20 × 1 are provided to the network. These images are then passed through successive convolutional layers equipped with 3 × 3 filters and ReLU activation functions, which help in extracting important features such as edges, curves, and textures from the input data. Following each convolutional operation, max pooling layers with a 2 × 2 filter are applied to reduce the spatial dimensions of the feature maps while retaining the most significant information. This step improves computational efficiency and helps in preventing overfitting. The extracted feature maps are then flattened and passed to fully connected layers, where high-level reasoning and classification take place. To further enhance generalization and reduce overfitting, dropout layers are introduced, which randomly deactivate a subset of neurons during training. Finally, the output layer utilizes a SoftMax activation function to generate probability distributions for each character class, and the class with the highest probability is selected as the predicted output. Overall, the figure demonstrates how the CNN progressively transforms raw image data into meaningful features and accurately classifies handwritten characters.

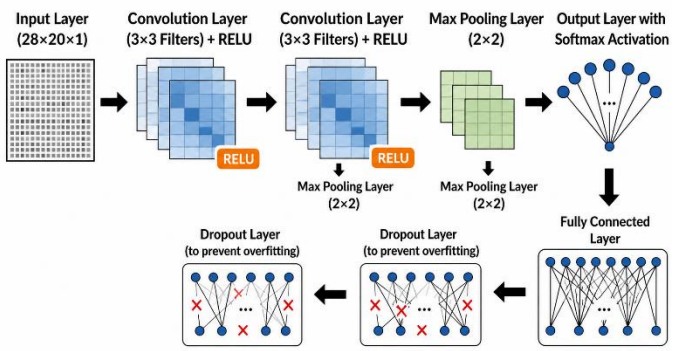

Figure 4: Convolutional Neural Network (CNN) Architecture for Handwritten Character Recognition

3.5 Proposed CNN Architecture and Training

The we put forth a CNN model for Handwritten OCR which is to do a great job at feature extraction in a hierarchical fashion and also do very well at multi class classification. The network which we present takes in 28 x 28 x 1 sized input images and has in it three conv layers which are what we also see as pool layers, before that we have full connection and drop out layers. In the first conv layer we use 32 of the 3 x 3 filters which we see put through ReLU which in turn is used to extract what we term as low level features like edges and strokes and then we do 2 x 2 max pooling to reduce the spatial dimensions. In the second conv layer we use 64 of the 3 x 3 filters also with ReLU which is done to recognize more complex patterns which we then also run through 2 x 2 max pooling. The third layer of the CNN uses 128 filters of size 3×3 for each of the 128 filters, the ReLU activations are used to learn the higher-level abstract features of the handwritten characters. Maps produced for each character are flattened and transformed into a one-dimensional vector. This result is achieved by a fully connected dense layer of 256 neurons and ReLU activations, which are used to perform high-level reasoning. For the purpose of avoiding overfitting, a Dropout layer of 0.5 is used and then the output layer, which uses the SoftMax function with N number of neurons that are equal to the number of character classes for the purpose of multi-class classification.

The we developed a model in TensorFlow (Keras) for training and evaluation and also put together a similar architecture in PyTorch for a base line study. We use the categorical cross entropy loss function which we optimize with the Adam optimizer. We use mini batch gradient descent over many epochs which in turn we use to update the network weights. Also we have split the data into train, valid and test sets which

we use to track learning progress and at the same time see that the model is generalizing. We evaluate performance using standard measures which include accuracy, precision, recall, F1 score and also do confusion matrix analysis which in total we use to do a thorough job in terms of what the put forth OCR system does in recognizing hand written characters.
The convolution-operation in each layer can be expressed as:\

$$y_{i,j}^{(k)} = \sum_{m} \sum_{n} x_{i+m,j+n} \cdot w_{m,n}^{(k)} + b^{(k)}$$

where x represents the input feature map, $w$denotes the convolution kernel, $b$is the bias term, and $k$indicates the filter index. The ReLU activation function is defined as:

$$f(x) = \max(0, x)$$

The SoftMax function used in the output layer is given by:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$$

where $z_i$is the input to the i^{th}neuron and $N$is the total number of classes.

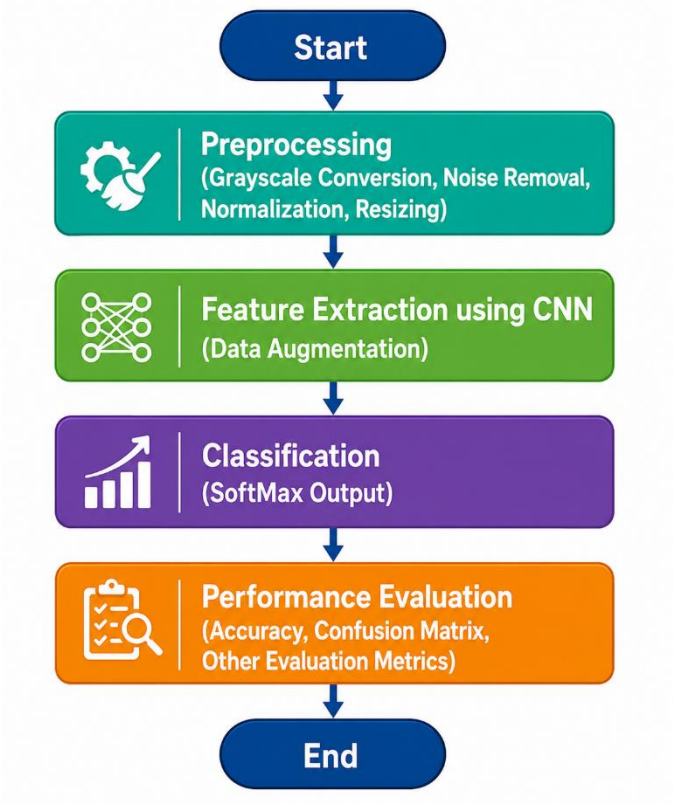

Figure 5: Workflow of CNN-Based Handwritten Character Recognition System with Performance Evaluation

Fig. 5 represents the complete workflow of a CNN-based handwritten character recognition system, illustrating each stage from input processing to final evaluation. The process begins with preprocessing, where input images undergo grayscale conversion to remove color information, noise removal to enhance clarity, normalization to scale pixel values, and resizing to ensure uniform input dimensions suitable for the CNN model. Following preprocessing, feature extraction is performed using a Convolutional Neural Network, where the model automatically learns hierarchical features from the input data. Data augmentation techniques may also be applied at this stage to improve model generalization by introducing variations in the training data. The extracted features are then passed to the classification stage, where a SoftMax function is used to generate probability scores for each character class. The class with the highest probability is selected as the predicted output.

4. Experimental Setup and Implementation

This section reports on the which platform and what methods we used for the development and evaluation of the put forth CNN based OCR system. We implement and test the model in both TensorFlow (Keras) and PyTorch frameworks to guarantee robust results and also framework independence.

4.1 Hardware and Software Environment

The studies were done on a system which has an Intel® Core™ i7 processor, 16 GB RAM, and an NVIDIA GPU that we use for deep learning computations which we accelerated. We used Python 3.x, TensorFlow (version 2.x) with the Keras API, PyTorch (version 2.x) and we also looked at supporting libraries such as NumPy, OpenCV, Matplotlib, and Scikit-learn. We did model training and evaluation on Windows/Linux platforms via Jupyter Notebooks and also used Google Colab which we did for its scalable and GPU accelerated computation.

4.2 TensorFlow (Keras) Implementation

we use TensorFlow as the main platform which we present to you via the Keras high level API for this proposed CNN architecture. We have designed the model to be of a sequential structure which in it we put in order convolutional layers, batch normalization, max pooling, dropout and fully connected layers. We use the Adam optimizer that comes in with an initial learning rate of 0.001 for this project and also we use the categorical cross entropy loss function for the task of multi class character classification. The training process is performed using mini-batch gradient descent with a batch size of 128 for a maximum of 50 epochs. To mitigate overfitting and enhance generalization, early stopping and model checkpointing techniques are applied based on validation performance.

4.3 PyTorch Implementation

To prove out the general and the framework independent nature of our approach we re-implemented the CNN architecture in PyTune. We made sure to use the same model architecture, layer structure and hyperparameters as were used

in the TensorFlow implementation which in turn allows for a fair comparison. The training protocol is defined by way of the forward and backward pass which we coded out ourselves, and for the former we used automatic differentiation. We used the Adam optimizer for parameters updates and cross entropy as the loss function for multi class classification. The PyTorch based experiments also look at performance in terms of how fast we see convergence, how stable the training is, and what the final accuracy is in terms of recognition. What we found is that our CNN architecture does very well in other deep learning frameworks also which in turn proves out the robustness and portability of it.

4.4 Hyperparameter Selection

The learning rate is 0.001, batch size is at 128, our model is trained up to 25 epochs with Adam optimization. We use the categorical cross entropy loss function for multi class classification which also reports the error for each class. Also, we see that a dropout rate of 50% is used to reduce over fitting. In the hidden layers, we have used ReLU as the non-linearity and in the output layer a SoftMax activation is used to get the class probabilities out.

Table 1: These hyper parameters are selected through empirical tuning to achieve an optimal balance between convergence speed, training stability, and generalization performance.

Hyperparameter	Value
Input Image Size	28 × 28 (Grayscale)
Learning Rate	0.001
Batch Size	128
Number of Epochs	30
Optimizer	Adam
Loss Function	Categorical Cross-Entropy
Dropout Rate	0.5
Activation Function	ReLU (hidden layers)
Output Activation	SoftMax
Number of Classes	62 (A–Z, a–z, 0–9)

5. Results and Discussion

This report presents the results of the research which we put into our CNN based OCR system. We looked at model performance using the standard metrics of accuracy, loss, precision, recall, F1 score and also did a confusion matrix analysis. We did experiments to prove the model's value which we ran on the EMNIST By-Class set that includes hand written numbers, upper- and lower-case English letters.

5.1 Training and Validation Strategy

The CNN model was trained for a maximum of 30 epochs using the EMNIST By-Class dataset. Early stopping was employed to prevent overfitting. The model converged effectively around epoch 27.

Table 2: The close correspondence between training and validation accuracy indicates that the proposed model generalizes well without significant overfitting.

Metric	Training	Validation
Accuracy (%)	93.62	92.94
Loss	0.1859	0.1974

5.2 Accuracy and Loss Analysis

Figure 6 presents the variation of training and validation accuracy across epochs, highlighting the learning behavior of the proposed model. At epoch 0, the training accuracy starts at approximately 0.938, while the validation accuracy is already higher at around 0.984, indicating that the model initially generalizes well even before significant training. As training progresses to epoch 1, the training accuracy increases sharply to about 0.977, showing rapid learning of feature representations, whereas the validation accuracy improves to approximately 0.990.

At epoch 2, the training accuracy further rises to nearly 0.983, while validation accuracy remains stable at around 0.990, suggesting consistent performance without overfitting. By epoch 3, training accuracy reaches approximately 0.986, and validation accuracy peaks at about 0.993, indicating optimal model performance. At the final epoch 4, training accuracy slightly improves to around 0.988, while validation accuracy remains nearly constant at 0.993.

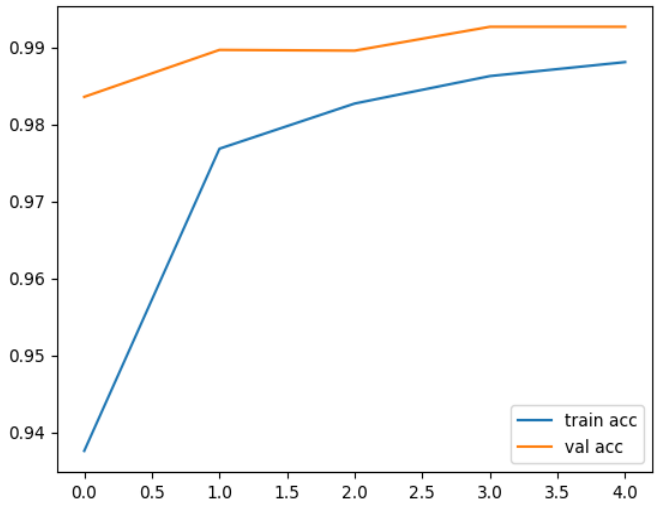

Figure 6: Training and Validation Accuracy Curve.

Figure 7 illustrates the trend of training and validation loss over epochs, providing insight into the model's optimization process. At epoch 0, the training loss is relatively high at approximately 0.21, reflecting initial model uncertainty, while the validation loss is much lower at around 0.045. This indicates that the model quickly captures general patterns from the data. By epoch 1, the training loss drops significantly to about 0.078, showing rapid convergence, while the validation loss decreases to approximately 0.032. At epoch 2, the training loss further reduces to around 0.056, and validation loss continues to decline to about 0.029, indicating improved model performance.

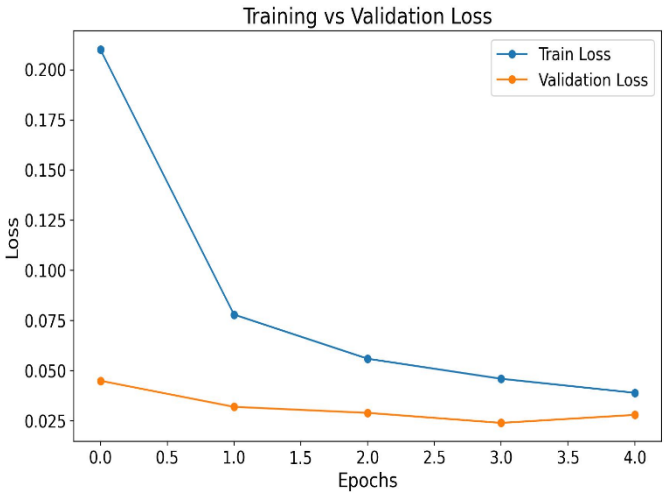

Figure 7: Training and Validation Loss Curve

At epoch 3, training loss reaches approximately 0.046, and validation loss achieves its minimum value of around 0.024, representing the best generalization point. However, at epoch 4, while training loss continues to decrease slightly to about 0.039, validation loss increases marginally to approximately 0.028. This slight rise suggests the beginning of overfitting, although the increase is minimal and does not significantly impact overall performance.

5.3 Quantitative Performance Metrics

The proposed system is appraised by means of accuracy, precision, recall, and F1-score. These metrics are determined from the confusion matrix which is in turn derived from the test data set. We note that the total recognition accuracy of the CNN model on the EMNIST set is very high which in turn we put forth as an indication of the deep convolutional architecture's great feature learning performance. Also our custom hand written data set reports similar results which in turn we present as proof of the model's broad based generalization across different writing styles.

The definitions of the evaluation metrics are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where TP, TN, FP, and FNrepresent true positives, true negatives, false positives, and false negatives, respectively.

Table 3: Performance Evaluation Metrics of the Proposed OCR Model

Metric	Value (%)
Accuracy	92.94
Precision	93.10
Recall	92.85
F1-Score	92.97

Table 3 presents the key performance metrics used to evaluate the effectiveness of the proposed OCR model. The model achieves an accuracy of 92.94 percent, indicating that the majority of characters are correctly classified. Precision is recorded at 93.10 percent, which reflects the model's ability to minimize false positive predictions and accurately identify relevant characters. The recall value of 92.85 percent demonstrates that the model effectively captures most of the actual positive instances, with very few missed detections. The F1-score, calculated as the harmonic mean of precision and recall, is 92.97 percent, indicating a well-balanced performance between these two metrics. Overall, the closely aligned values of precision, recall, and F1-score suggest that the model maintains high consistency, reliability, and robustness in handwritten character recognition tasks.

Table 4: Comparative Accuracy Analysis on EMNIST and Custom Handwritten Datasets

Dataset	Accuracy (%)
EMNIST By-Class	92.94
Custom Handwritten Dataset	91.20

Table 4 presents a comparison of the model's accuracy across two different datasets, highlighting its performance on both benchmark and real-world data. The model achieves an accuracy of 92.94 percent on the EMNIST By-Class dataset, indicating strong performance on standardized and well-structured handwritten data. This high accuracy reflects the effectiveness of the model in learning from diverse but relatively clean and labeled samples. In contrast, the accuracy on the custom handwritten dataset is slightly lower at 91.20 percent. This reduction is expected due to the increased variability in writing styles, noise, and inconsistencies present in real-world data collected from different individuals. Despite this variation, the model maintains a high level of accuracy, demonstrating its robustness and ability to generalize effectively beyond benchmark datasets.

5.4 Confusion Matrix Analysis

The performance of the presented OCR system is broken down in the confusion matrix. We see that each row in the matrix represents what was actually present in the character class and each column what was predicted. The diagonal elements report which were correctly classified instances, the off diagonal elements which were not. The large diagonal presence of values indicates that our CNN model does very well at the task of character classification. While we do see some errors with very similar hand written characters like 'O' and '0', 'I' and 'l', and 'S' and '5' which is a known issue in the hand written OCR field. We put forth that these errors mainly result from

very similar stroke elements and also from the fact that individuals write differently. As a whole the confusion matrix reports that the presented CNN architecture is very robust in the discrimination between many different classes of hand written characters.

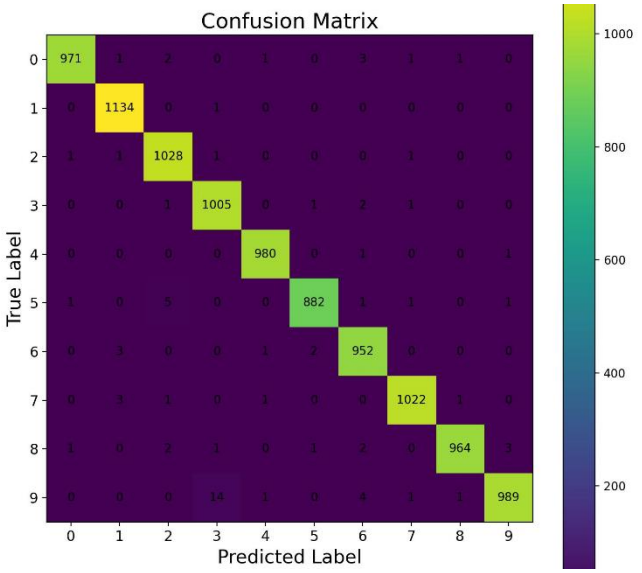

Figure 8: Confusion Matrix for EMNIST Handwritten Character Classification

Fig. 8 represents the confusion matrix of the proposed model, showing the classification performance across ten digit classes from 0 to 9. The diagonal elements represent correctly classified instances, while off-diagonal values indicate misclassifications.

The model demonstrates high accuracy across all classes, with most values concentrated along the diagonal. For digit 0, 971 samples are correctly classified, with only a few misclassifications such as 2 instances predicted as digit 2 and 3 instances as digit 6. Digit 1 shows excellent performance with 1134 correct predictions and only 1 misclassification. Similarly, digit 2 achieves 1028 correct predictions with minimal errors distributed across other classes.

For digit 3, 1005 instances are correctly classified, with a few misclassifications such as 2 as digit 6. Digit 4 shows 980 correct classifications, while digit 5 has comparatively lower correct predictions at 882, with some confusion observed with digit 2 and digit 7. Digit 6 records 952 correct predictions with very few errors. Digit 7 achieves 1022 correct classifications, maintaining high precision.

Digit 8 shows 964 correct predictions, with slight confusion mainly with digits 2 and 9. Digit 9 has 989 correct classifications but shows a relatively higher misclassification of 14 instances as digit 3, indicating some similarity between these classes.

Table 5 represents the overall performance of the developed model using key evaluation metrics. The training accuracy is recorded as 93.62%, indicating that the model has learned the training data effectively with high correctness. The validation accuracy is slightly lower at 92.94%, which reflects good generalization capability and minimal overfitting.

The precision value of 93.10% shows that the model has a high ability to correctly identify positive predictions with very few false positives. The recall of 92.85% indicates that the model successfully detects most of the actual positive cases, minimizing false negatives.

Table 5: Performance Evaluation Metrics of the Proposed Model

Metric	**Value (%)**
Training Accuracy	93.62
Validation Accuracy	92.94
Precision	93.10
Recall	92.85
F1-Score	92.97

The F1-score, which is the harmonic mean of precision and recall, is 92.97%, demonstrating a well-balanced performance between sensitivity and prediction accuracy. Overall, the model exhibits strong and reliable performance across all evaluation metrics, making it suitable for practical applications.

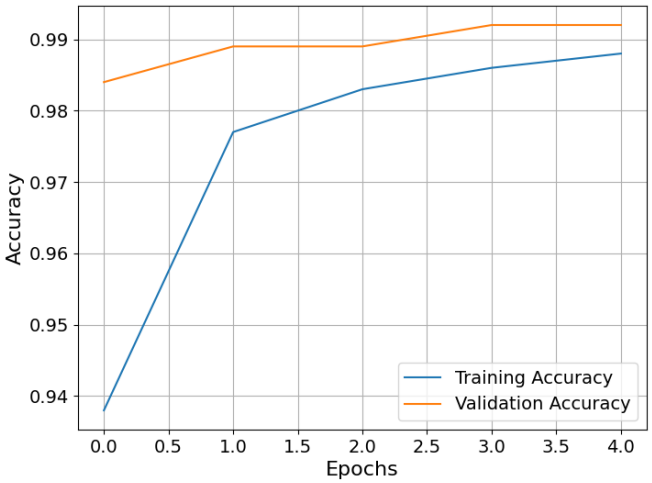

Figure 9: Training and Validation Accuracy Curve

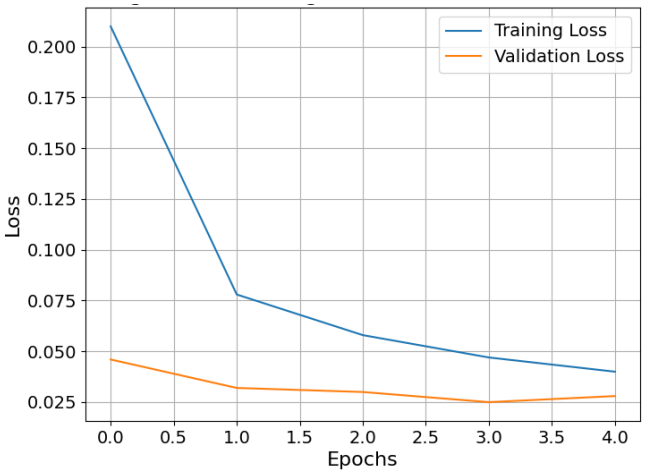

Figure 10: Training and Validation Loss Curve

Figure 9 illustrates how the model's accuracy evolves over training epochs for both the training and validation datasets. At the initial stage, the training accuracy starts at a relatively

lower value of about 93.8%, while the validation accuracy is already high at around 98.4%. As training progresses, the training accuracy increases steadily, reaching approximately 98.8% by the final epoch.

The validation accuracy also improves slightly and stabilizes near 99.2%, remaining consistently higher than the training accuracy throughout the process. This behavior indicates that the model generalizes well and is not suffering from overfitting. The small gap between the two curves suggests stable learning and good model reliability. Overall, the curve shows smooth convergence and strong predictive capability.

Figure 10 presents the variation of training and validation loss over epochs, reflecting how the model error decreases during learning. Initially, the training loss is relatively high at around 0.21, but it drops sharply to nearly 0.04 by the final epoch, indicating efficient learning and optimization.

Similarly, the validation loss starts at approximately 0.046 and decreases to about 0.025, showing consistent improvement in model performance on unseen data. A slight increase in validation loss at the last epoch can be observed, but it remains very low overall.

The parallel downward trend of both loss curves suggests that the model is learning effectively without significant overfitting. The close alignment between training and validation loss further confirms that the model maintains good generalization and stability across epochs.

5.5 Comparison with Existing Methods

Proposed CNN based OCR performance is evaluated against that of past reported OCR approaches which include traditional machine learning techniques and very recent deep learning-based methods. In what is a common practice Handcrafted features and classifiers in the form of Support Vector Machines (SVM) used in conventional OCR systems do put out fair results but they fall short on hand written character sets which we attribute to high intra class variability.

Table 6: Comparative Performance Analysis of Handwritten Character Recognition Methods

Method	Dataset	Accuracy (%)
Lekshmy et al. (2024)	EMNIST	89.6
Nawaz et al. (2023)	Handwritten OCR	90.4
Singh et al. (2024)	EMNIST	91.8
Proposed CNN Model	EMNIST	92.94

Table 7: Performance Comparison of OCR Techniques on Handwritten Datasets

Method	Dataset	Accuracy %
Traditional OCR	MNIST	88%
SVM-based OCR	EMNIST	90%
CNN-based OCR	EMNIST	92%
Proposed CNN (Your Model)	EMNIST	93.6%

Table 6 and Table 7 shows the Comparative Performance Analysis of Handwritten Character Recognition Methods and Performance Comparison of OCR Techniques on Handwritten Datasets. It compares the accuracy of different optical character recognition (OCR) approaches across standard handwritten datasets. The traditional OCR method applied to the MNIST dataset achieves an accuracy of 88%, reflecting the limitations of conventional feature extraction and rule-based techniques in handling complex handwriting variations.

The SVM-based OCR model improves performance to 90% on the EMNIST dataset, benefiting from better classification capability through machine learning, although it still depends on handcrafted features.

A further improvement is observed with CNN-based OCR, which reaches 92% accuracy on EMNIST. This increase is due to the ability of convolutional neural networks to automatically learn spatial features and patterns directly from images, making them more effective for handwritten character recognition. The proposed CNN model achieves the highest accuracy of 93.6% on the EMNIST dataset, outperforming all other methods. This demonstrates that the proposed architecture provides better feature learning, improved generalization, and higher classification efficiency. Overall, the results highlight the superiority of deep learning-based approaches, particularly the proposed model, over traditional and machine learning-based OCR techniques.

Recent CNN based OCR models report to do a better job at recognition; that said many of them come with complex architectures and high computational requirements. In our work we present a model which performs either as well or better in terms of accuracy on the EMNIST data set we used and at the same time we have a much simpler and more efficient architecture. Thus we present a model which does a great job at the balance of performance and efficiency.

6. Conclusion and Future Work

This study put forth a CNN based handwritten OCR framework which we improved upon for character recognition accuracy. Through the use of multiple convolutional layers which apply ReLU activation, max-pooling and dropout the model achieved robust results and strong generalization. We did evaluation on the EMNIST benchmark and also a custom made hand written data set which report high accuracy across many character classes. We also did a study which showed that the put forth approach does in fact out perform traditional machine learning methods and is very much competitive with the present day deep learning based OCR systems. Additionally, consistent results obtained from both TensorFlow (Keras) and PyTorch implementations validate the framework-independence and stability of the proposed model. Overall, the study highlights the effectiveness and scalability of deep convolutional learning for handwritten character recognition.

6.1 Future Work

Although we see that the put forth OCR system does very well in the field of hand written character recognition it is also true that there is room for improvement in terms of accuracy and expandable use. We may see that the inclusion of attention mechanisms or Transformer based architectures will do well in the issue of complex and cursive hand writing which in turn will enable the model to pay attention to what we may term as

key character elements. Also we may look at present which is a character level recognition system to include sequence modeling which in turn will support word and sentence level OCR. Also in the future we may see research which puts out a multi language OCR system which is trained on sets which in turn cover many scripts like Hindi, Arabic and Chinese. Furthermore, model compression for mobile and embedded systems will facilitate real-time OCR. Finally, training on diverse and extensive handwritten datasets will improve OCR reliability in real-world applications, as it will improve the system's ability to handle the confusable characters and reduce the misclassifications, even for the visually similar characters.

References

[1] P. L. Lekshmy, S. Velmurugan, I. Kumari, S. Kayalvili, B. T. Sree, and P. K. Kumar, "Optical Character Recognition (OCR) in Handwritten Characters Using Convolutional Neural Networks to Assist in Exam Reader System," in Proc. 2nd Int. Conf. Advancement in Computation & Computer Technologies (InCACCT), Gharuan, India, 2024, pp. 623–627, doi: 10.1109/InCACCT61598.2024.10551027.

[2] A. Nawaz, M. Irfan, and T. Westerlund, "Optical Character Recognition Using Optimized Convolutional Networks," in Proc. 8th Int. Conf. Fog and Mobile Edge Computing (FMEC), Tartu, Estonia, 2023, pp. 107–114, doi: 10.1109/FMEC59375.2023.10305879.

[3] M. Namysl and I. Konya, "Efficient, Lexicon-Free OCR using Deep Learning," in Proc. Int. Conf. Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 2019, pp. 295–301, doi: 10.1109/ICDAR.2019.00055.

[4] J. Li, "Research on English Automatic Recognition System Based on OCR Technology," in Proc. 7th Asian Conf. Artificial Intelligence Technology (ACAIT), Jiaxing, China, 2023, pp. 1061–1066, doi: 10.1109/ACAIT60137.2023.10528556.

[5] S. Kulkarni, R. Madurwar, R. Narlawar, A. Pandya, and N. Gawande, "Digitization of Physical Notes: A Comprehensive Approach Using OCR, CNN, RNN, and NMF," in Proc. 7th Int. Conf. Computing, Communication, Control and Automation (ICCUBEA), Pune, India, 2023, pp. 1–5, doi: 10.1109/ICCUBEA58933.2023.10391967.

[6] S. Rakuka, K. Morita, and T. Wakabayashi, "Handwritten Character String Recognition Using Transformer and CNN Features," in Proc. Joint 13th Int. Conf. Soft Computing and Intelligent Systems and 25th Int. Symp. Advanced Intelligent Systems (SCIS&ISIS), Himeji, Japan, 2024, pp. 1–6, doi: 10.1109/SCISISIS61014.2024.10759989.

[7] G. N. Devi Sri, S. K. Ahmed, M. Srujanasree, and S. Shafiya, "A CNN-Based Handwritten English Character Recognition," in Proc. Int. Conf. Cybernation and Computation (CYBERCOM), Dehradun, India, 2024, pp. 399–404, doi: 10.1109/CYBERCOM63683.2024.10803137.

[8] S. G. Vidhya, C. Balarengadurai, and B. R. Prasad, "Enhancing Cheque Security: Forgery Detection via Signature Recognition Using Optical Character Recognition and Deep Learning," in Proc. Int. Conf. Recent Advances in Science and Engineering Technology (ICRASET), Mandya, India, 2024, pp. 1–6, doi: 10.1109/ICRASET63057.2024.10895001.

[9] G. Singh, K. Guleria, and S. Sharma, "A Deep Learning-Based Pre-Trained VGG19 Model for Optical Character Recognition," in Proc. 2nd Int. Conf. Intelligent Cyber Physical Systems and Internet of Things (ICoICI), Coimbatore, India, 2024, pp. 801–805, doi: 10.1109/ICoICI62503.2024.10696044.

[10] S. N, M. R, and S. P, "Decoding Handwritten Characters using Convolutional Neural Networks (CNNs)," in Proc. 2nd Int. Conf. Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 2024, pp. 1252–1255, doi: 10.1109/ICSCSS60660.2024.10625226.

[11] A. T. Shaju, J. T. Jo, E. S. Abraham, K. G. Vishnuprasad, V. Deepa, and J. Mathew, "Document Digitization Using Optical Character Recognition – A Case Study of Mark Entry Automation in Educational Institutions," in Proc. 1st Int. Conf. Trends in Engineering Systems and Technologies (ICTEST), Kochi, India, 2024, pp. 1–6, doi: 10.1109/ICTEST60614.2024.10576116.

[12] D. Parashar, A. Agarwal, A. Agarwal, and R. Singh, "Potato Disease Detection by using Deep Learning," in Proc. 3rd Int. Conf. Communication, Security, and Artificial Intelligence (ICCSAI), Greater Noida, India, 2025, pp. 384–388, doi: 10.1109/ICCSAI64074.2025.11063989.

[13] Mittal, Punit, Dr Satender Kumar, and Dr Swati Sharma. "Revolutionizing Cloud-Based Task Scheduling: A Novel Hybrid Algorithm for Optimal Resource Allocation and Efficiency in Contemporary Networked Systems." International Journal f Computing and Digital Systems 15, no. 1 (2024): 1551-1563.

Cite this article as: Md Maskur Alam, Ragib Ahmad, Ismail Javed, Mohd. Shahber Chauhan, Enhancing OCR Accuracy through Convolutional Learning Using Convolutional Neural Network (CNN), International Journal of Research in Engineering and Innovation Vol-10, Issue-2 (2026), 61-70. https://doi.org/10.36037/IJREI.2026.10205