Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Optical Character Recognition Systems for Handwritten Text

Repsong Lepcha, Sanidhya Shekhar, Darpan Dahal
Department of Computer Engineering, SIST, Sikkim, India

Dibya Thapa
Assistant Professor, Department of Computer Engineering, SIST, Sikkim, India

Abstract - ***Optical Character Recognition (OCR)*** is a technology used to convert images of text into machine-readable form. Even though OCR systems have made significant progress in reading printed text, handwriting recognition remains a challenge, specifically if its untidy or cursive and when the quality of image is poor and not bright enough. In this project, we address these issues by considering them as limitations of OCR systems. Several previous studies are not well suited for real-time applications, as they depend on complex, large-scale model architectures and relatively small datasets. In order to address this, our project first tested the performance of lightweight Machine Learning classifiers on handwritten characters, including Support Vector Machine (SVM), Random Forest, Extreme Gradient Boosting (XGBoost), CatBoost, and K-Nearest Neighbors (k-NN). These machine learning models offered a moderate accuracy and functioned as baseline classifiers. However, the Machine Learning models displayed limitations because handwriting styles vary widely.To improve performance we implemented a Deep Learning model based on Convolutional Neural Networks (CNN) that automatically learns features from raw images. Outperforming all Machine Learning classifiers, the CNN attained the highest accuracy of 98%. The system was tested on a combination of Extended Modified National Institute of Standards and Technology (EMNIST) samples and real-world handwritten data, including messy and challenging handwriting and produced a reliable output.

Keywords: OCR, Machine Learning, Deep Learning, CNN, EMNIST.

1. INTRODUCTION

OCR is a technology that transforms text images into digital formats that can be edited and searched. Today, OCR is helping not only in digitizing the handwritten manuscripts , but also helps in converting the typewritten documents into digital form [1] . Although OCR systems are excellent at identifying text that has been printed, they still struggle with handwritten text. The size, shape, and style of a person's handwriting can vary. OCR tools find it more difficult to read images with poor quality, poor lighting, and messy or cursive writing.

Most modern OCR systems rely on large datasets and complex deep learning models that require significant computational power. This functionality is similar to how Google Lens works on the Google Search and Photos apps on iOS [2]. The use of OCR technologies is expanding across a number of industries, including document digitization, education, healthcare, and automation tasks like reading handwritten notes or forms.

Many OCR systems rely on cloud-based processing and are not designed for offline environments. The project's objective is to develop an efficient OCR system that can recognize different handwriting styles. The automatic recognition of handwritten text can be extremely useful in many applications where it is necessary to process large volumes of handwritten data, such as recognition of addresses and postcodes on envelopes, interpretation of amounts on bank checks, document analysis, and verification of signatures [3]. This makes them less suitable for lightweight or offline systems that do not rely on cloud servers.

As the most difficult problem in the field of OCR is recognition of unconstrained cursive handwriting [4], the aim of this project is to create a fast and easy to use OCR system that can recognize handwritten text.The project first used lightweight machine learning classifiers as baseline models to test. But the machine learning models did not achieve the desired level of accuracy due to significant inconsistency in handwritten styles. A CNN model was employed for better performance on the handwritten data where features are learned automatically. Experiments were conducted on real data with different handwriting styles to evaluate the system performances.

2. LITERATURE REVIEW

OCR is a technology that creates machine-readable text from images of handwritten or printed text.From pattern matching systems to

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

machine learning that is better at recognizing handwriting, and over time to deep-coverage models such as CNN with higher accuracy on handwritten text OCR has made significant strides.

C.K. Gomathy et al. in their paper "Optical Character Recognition" (2022) explained how OCR works and how it's used in tools like Google Lens and Apple's Live Text. It shows how important preprocessing and segmentation are used to improving OCR accuracy[1]. Ameer Majeed and Hossein Hassani, "Ancient but Digitized: Developing Handwritten Optical Character Recognition for East Syriac Script Through Creating KHAMIS Dataset," 2024. It developed OCR techniques to digitize and preserve the ancient Syriac script, introduced the KHAMIS dataset, the first dataset for East Syriac handwritten script and tested various machine learning algorithms on the KHAMIS dataset to assess recognition accuracy. However, a number of recent studies demonstrate that deep learning techniques perform better than machine learning models for challenging handwriting tasks [2].

Mahmoud SalahEldin Kasem et al. "Advances and Challenges in Arabic Optical Character Recognition: A Comprehensive Survey" (2023), despite its emphasis on Arabic OCR, this paper addressed common handwriting problems like cursive writing, different character shapes, and small datasets that also affect English handwriting [3]. Syed Sajid Ullah et al. in their paper "Handwritten Digit Recognition: An Ensemble Approach For State Of The Art Performance" (March 2025) applied CNN to extract features and then used each feature to display how deep learning models increase accuracy for the handwritten data [4].

Avinash Malladhi et al. in their work "Transforming Information Extraction: Oracle AI & Machine Learning Techniques Applied to OCR Systems" (2023) describe how cursive writing & poor image quality were some of the challenges for OCR systems[5].

Dibya Thapa and Rebika Rai in their work "FREQ-EER: A Novel Frequency-Driven Ensemble Framework for Emotion Recognition and Classification of EEG Signals" (2 October 2025) highlighted the effectiveness of different augmentation techniques, when combined with effective feature extraction achieving high accuracy with low computational cost, which motivates the use of different augmentation techniques in the proposed OCR system [6].

Jamshed Memon et al. complied "Handwritten Optical Character Recognition: A Comprehensive Systematic Literature Review" (2020).This paper reviewed over 140 research papers and showed how OCR moved from rule-based methods to machine learning models like SVM, k-NN, Decision Tree, and Random Forest. Additionally, it clarified issues like overlapping characters and different writing styles. This was important for us to recognzie the limitations of "classical" machine learning classifiers which we regarded as our baseline models before using CNN [7].

Through literature review, we were able to find out that although the classical Machine learning models such as SVM, k-NN and Random Forest give acceptable results their performance are suboptimal on high handwriting variation. It has also been demonstrated in literature that CNN produce much higher accuracy for handwritten character recognition based on its processing with an automatic feature learning mechanism. Right preprocessing, feature engineering and model selection are the key to higher accuracy.

3. MATERIAL AND METHODOLOGY

The procedure for the implementation of the proposed handwritten text OCR system is explained in this section. To improve performance, the project initially applied traditional machine learning classifiers as baseline models

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

before using a CNN. The methodology includes several stages, starting with data collection, preprocessing, and feature extraction. It proceeds to model training, testing, and evaluation [3]. Each stage was designed to keep the system lightweight while ensuring accuracy. Extracting text from documents or images into machine-readable text using OCR technology involves several key steps [12]. Both the machine learning baseline models and the final CNN model used the same dataset and preprocessing steps. The overall view of the model is presented in the block diagram (Figure 1).

Dataset	Language	Content	Size
MNIST	English Digits	Handwritten Digits (0-9)	70,000 Letters
EMNIST	English Alphabet	Handwritten Letters and Digits	8,10,000 Letters
IAM	English Words	Handwritten Words	1,50,000 Letters

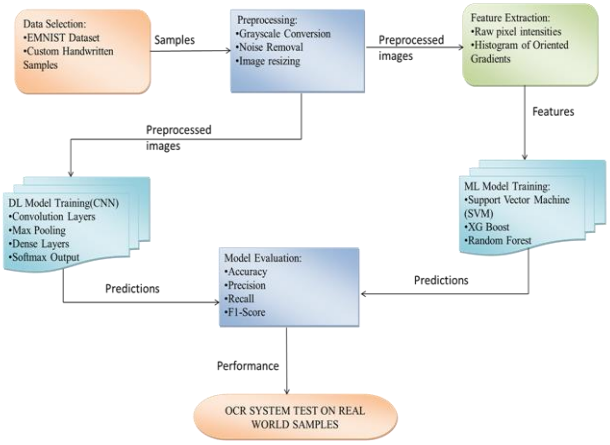

Figure 1. Handwritten Text Recognition

3.1 DATA COLLECTION

The data collection step is the phase in which all raw data intended for training and testing the model are gathered. In image analysis, this typically involves training datasets of images that depict characters to be recognized. For training and testing the handwritten character recognition model, combinations of a standard dataset were used. The main dataset was Extended Modified National Institute of Standards and Technology (EMNIST), which is an extended version of the popular Modified National Institute of Standards and Technology (MNIST) dataset. Unlike MNIST that only has digits, EMNIST also contains handwritten alphabets (A–Z), both uppercase and lowercase. Along with this, a custom dataset of handwritten alphabets was created. These were collected by asking volunteers to write characters on paper, then scanning and saving them as images. This step was done to test whether the model can handle different writing styles instead of just performing well on a standard dataset.. The collected dataset was used for training both machine learning classifiers and the final CNN model. Available Dataset for OCR is shown in (Table 1):

Table 1. Dataset for OCR

3.2 DATA PREPROCESSING

Data preprocessing refers to the steps taken to clean and prepare raw data so that it can be used effectively by a model [6]. Technique such as resizing, noise removal, and grayscale were used. First, all images were converted into grayscale. In order to enhance the images, noise reducing techniques such as Gaussian or median filter were used to remove background marks. All images were then resized to 28×28 pixels for consistency. After that, the pixel values were normalized which helps the model train faster and more accurately. Data augmentation methods such as rotation, scaling, shifting, and flipping were used. For CNN training, preprocessing included only resizing, normalization, and data augmentation since CNN learns features automatically from images.

3.3 FEATURE EXTRACTION

Feature extraction is the process of converting raw data into a set of measurable characteristics that can be understood by machine learning models

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

[7]. The first method was using raw pixel intensities, where the pixel values of the image were directly flattened into a feature vector. Although this is simple, it does not capture much structure. For Machine Learning models, features such as raw pixel intensities, Histogram of Oriented Gradients (HOG), and zoning were extracted to represent each character. For the CNN model, no manual feature extraction was required because the convolution layers automatically learn edge patterns, shapes, and high-level features from the images. By combining these feature extraction methods, models with both detailed and general information about each character was provided.

3.4 MODEL TRAINING

Model training includes two phases: training baseline machine learning classifiers and training the final CNN-based Deep Learning model. The model used these input features to learn patterns and form a mechanism to make predictions on new, unseen data. After data preparation, samples were then split into training and test sets with 80% used in the training and the other 20% in the testing phase. In this manner, the model is trained on a majority of the data but tested on unseen values. Then, for the training of the machine learning models including Support Vector Machine, Random Forest and XGBoost. After analyzing machine learning classifiers, we trained a CNN with the preprocessed images.

3.5 MODEL EVALUATION

After training the model, it is important to check its performance on test data. The first metric used is accuracy, which will give us the percentage of characters that the model predicted correctly. Precision tells how many of the characters predicted were actually right, while recall tells how good the model is at finding all of the actual characters. The F1 score, instead, is a combination of precision and recall making it a single averaged measure of trade-off between the two. We also used a confusion matrix, which is similar to a table that compares the predicted and actual labels. This allows us to more readily observe which characters the model identifies correctly, and with which it gets mixed up. All evaluation measures were computed for both machine learning and the CNN model, with the top accuracy score from CNN.

3.6 MACHINE LEARNING CLASSIFIER

4. Machine Learning classifiers plays important role in the development of the handwritten character OCR system. Since they help in the automatic classification of an input character image into the correct textual classes. These classifiers have the ability to precisely identify unseen handwritten characters despite the differences in writing, size, and orientation. The need to improve the overall accuracy of the system is the reason why supervised learning models have been incorporated in this work.

3.6.1 K-Nearest Neighbors (k-NN)

k-NN is a simple and easy to understand classification algorithm defined by equation (1) and (2). It does this by looking at the k closest data points simplifying the model near where we want to classify new points and determining how many of those points are labeled one way versus the other, then classifying a new sample from that knowledge. To determine the distance, it typically employs the Euclidean distance formula to find out how far apart points are from each other in that feature space. Since the technique only requires to store the training data and compare distances at prediction time, it is usually referred as a "lazy learner.

$$\hat{y} = \text{mode}(y_i \mid i \in N_k(x)) \quad (1)$$

Where:

- $\hat{y}$ is the predicted label.
- $N_k(x)$ is the set of k nearest neighbors to point x.
- Distance metric**:** usually Euclidean distance:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

$$d(x, x_i) = \sqrt{\sum_{j=1}^{n}(x_j - x_{ij})^2} \quad (2)$$

3.6.2 Support Vector Machine (SVM)

SVM is a powerful classification algorithm defined by equation (3) that attempts to find optimal boundary, called hyperplane that distinguishes between two classes with the largest margin. In other words, it attempts to draw a line or higher-dimensional plane such that the classes are as far away from each other as possible. It can also deal with non-linear data though the kernelized form like linear, polynomial or RBF kernel functions that can enable SVM transform the data into a space where separation is really simple. The last prediction is decided by the side of hyperplane.

$$f(x) = \text{sign}(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b) \quad (3)$$

Where:

- α_i are Lagrange multipliers.
- y_i are labels.
- $K(x, x_i)$ is a kernel function (linear, polynomial, RBF, etc.)
- b is the bias term.

3.6.3 Extreme Gradient Boosting (XGBoost)

XGBoost is an advanced ensemble machine learning technique defined by equation (4) that groups several decision trees together to form a strong classifier. In contrast to Random Forests, where trees are constructed independently, XGBoost build trees successively and every new tree is designed to reduce the errors of previous ones. This process is called boosting. The algorithm is also based on gradient descent to gradually reduce the error, so it can be very efficient and accurate. Given its speed and performance, XGBoost is frequently employed in machine learning competitions and used in real world applications.

$$\hat{y}_i = \sum_{m=1}^{M} f_m(x_i), \; f_m \in F \quad (2)$$

Where:

- F is the space of decision trees.
- f_m is a decision tree model.

3.6.4 Random Forest

Random Forest is another ensemble method defined by equation (5) that builds a larg (5) ber of decision trees and combines their results in order to do the final prediction. Single tree is learnt on a subsample of the data and features to avoid over-fitting and promote diversity among trees. With all trees constructed, the final output is determined based on a majority vote in the case of classification or an average in the case of regression. Such randomness and combining multiple trees leads to the high accuracy, stability, and robustness of a Random Forest over a single decision tree.

$$\hat{y} = \text{mode}(h_1(x), h_2(x), \ldots, h_N(x))$$

Where:

- $h_i(x)$ is the prediction of the ith decision tree.
- Final prediction is the majority vote from all trees.

3.6.5 CatBoost

CatBoost is a gradient boosting model like XGBoost defined by equation (6), however it's designed for categorical data and doesn't need too much feature engineering as one-hot encoding. It works well in preventing overfit using some advanced methods such as ordered boosting and data random permutation. As with all other boosting techniques, CatBoost builds trees in series, such that each successive tree corrects the errors of previous ones. A key advantage of this

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

method is its strong performance on datasets with a substantial number of categorical features, while minimizing the effort required to select an appropriate polynomial degree.

$$\hat{y}_i = \sum_{m=1}^{M} f_m(x_i) \quad (6)$$

Here,

- $\hat{y}_i$ is the predicted value for the i-th sample.
- M is the total number of trees (iterations of boosting).
- $f_m(x_i)$ is the prediction from the m-th decision tree for the input x_i.

3.7 DEEP LEARNING CLASSIFIER

5. Deep Learning classifiers include advanced forms of machine learning, where classifiers automatically learn feature representations of raw input data. Deep learning classifiers, such as CNN, have been used in handwritten character recognition systems, enabling the achievement of high accuracy in Optical Character Recognition. Deep learning classifiers outperform other classifiers because of the ability of the classification algorithms to keep up with changes in writing, noise, etc.

3.7.1 Convolutional Neural Network

The CNN was selected as the deep learning model in this work instead of machine learning classifiers due to its best performance. CNN is a kind of network which is widely used for images. The primary feature of CNN is it automatically does the work to discover important features from images without us manually having to do that for machine learning models with HOG or pixel intensities. CNN consists of multiple layers, and each layer serves a specific function.The main layers used in our model are explained below:

3.7.1.1 Convolution Layer

The convolution layer applies small filters over the image to detect patterns like edges, curves, and corners as defined by equation (7). These filters move across the image and highlight important parts of the handwriting. As we go deeper into the network, the filters learn more complex shapes and strokes. This layer helps the CNN automatically extract features without manual work. It is the main layer responsible for understanding the structure of charact (6)

$$Z = W * X + b$$

Where:

- W = filter
- X = input image
- b = bias
- Z = output feature map

3.7.1.2 ReLU Activation Function

After the convolution layer, ReLU is introduced to eliminate negative values and retain positives as defined by equation (8). This allows the model to learn more quickly and to understand more intricate patterns. ReLU also solves the issue of vanishing gradients which can halt training. It essentially makes the network pay attention to what's relevant. This simple function enhances the accuracy and speed. It is just that it replaces negative values with 0:

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

This helps the model learn more complicated structures.

3.7.1.3 Max Pooling Layer

The purpose of the max pooling is to decrease the dimensionality of the feature maps by simply retaining only maximum value within some small region. This process enables filtering out too many irrelevant details, while trying to recover important ones. Pooling helps the model to be faster and more resistant to overfitting. It also enables the CNN to recognize characters even when handwriting is off a little. In general, it

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

simplifies the feature maps without discarding essential information.

3.7.1.4 Flatten Layer

The flatten layer changes the 2D feature maps into a 1D vector. This is necessary, because dense layers can only process one long vector. Flattening is used to transition from the feature extraction section of the network to the decision section. It learns nothing, it simply arranges the data in a sensible order. This stage readies the features for classification.

3.7.1.5 Fully Connected (Dense) Layers

The dense layer receives all the features detected and tries getting to guess what character it is. It takes edges, shapes, and patterns to come up with a final answer. This layer will learn which are the important features for each class as defined by equation (9). It makes it easier to differentiate between similar-looking characters. The latter layer is vital to the last step of classificatic (9)

$$h = W^T x + b \quad (9)$$

Where:

- x = input from the flatten layer
- W = weights
- b = bias

3.7.1.6 Softmax Output Layer

The last layer is a softmax that transforms the final outputs into class likelihoods as defined by equation (10). It tells the model how certain which character is correct based on learned features. The class that has the highest probability is taken as the final prediction. By comparing the results of Softmax and SVM, we can find out the classification ability against different level categories by these two loss functions: on multi-class, softmax is trained better with similar performances to svm. It means that softmax is not only good at multi-class problem but also more useful in A–Z recognition.

$$\hat{y}i = \frac{e^{zi}}{\sum_{j=1}^{K} e^{zi}} \quad (10)$$

Where:

- K = number of classes
- ŷi= probability of class i

6. RESULT AND DISCUSSION

After training and testing on the prepared dataset, we measured the performance of each classifier by looking at accuracy and improvements from preprocessing. The (Table 2) summarizes the results for the CNN, SVM, Random Forest, XGBoost Ensemble, CatBoost, and k-NN model.

4.1 CLASSIFICATION

According to the results of the machine learning classifiers including SVM achieved an accuracy of 88% while Random Forest and CatBoost achieved an accuracy of 91%. k-NN achieved similar results, with 90% accuracy after preprocessing. The performance of the XGBoost Ensemble model was definitely better than all single ML classifiers with highest accuracy of 96.87 %, which had achieved pretty good traditional machine learning mode ls in our experiment. However, after using the CNN model, it's performance enhanced even more. Compared with the ML model which relies on manual feature extraction, CNN can automatically extract features from images and achieve better performance in handwritten character recognition. Because the CNN model ended with an accuracy of 98%, this was also a bit more accurate than every ML-based method. This shows that deep learning worked well in distinguishing handwritten characters than the conventional machine learning techniques. The

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

accuracy of each model is summarized in the bar graph as shown in (Figure 2), where CNN achieves the highest performance compared to traditional machine learning methods. The (Table 2) below also shows the accuracy we got for each model when tested on the data:

Table 2. Accuracy achieved using various classifiers

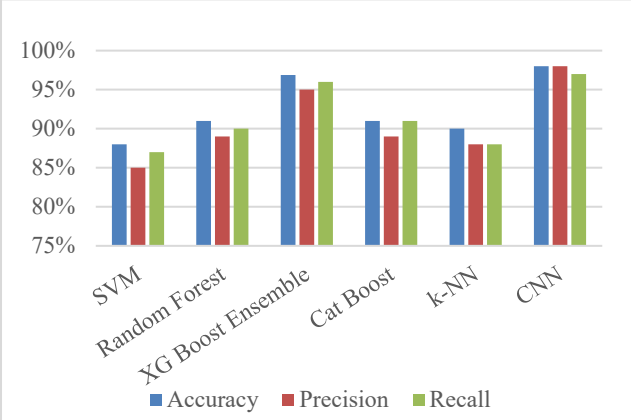

Figure 2. Accuracy achieved using various classifiers

4.1.1 Receiver-Operating Characteristic (ROC Curve)

The ROC plots show how well each model tells apart hand-drawn letters. A line closer to the upper-left corner means stronger results. Most lines sit tight to that spot, so most models work quite well. Alongside the machine learning ones, the CNN stands out its Area Under the Curve (AUC) is almost 1, its curve nearly touches the upper-left edge. This represents that the model is separating the classes of letters accurately with a very low rate of false positives. The proximity of the curves near the top-left corner of the graph represents that the classes are performing very consistently. The performance of the SVM model is also excellent, with micro and macro AUCs ranging approximately from 0.996 to 0.994 as shown in (Figure 3). The Random Forest model displays a sharp increase near the y-axis, and the micro-average and macro-average AUC values are close to 0.99 as shown in (Figure 4). The graph of the ROC curve is very close to the top-left corner of the graph. It is an indication of a highly reliable classifier. Compared with the performance of the Random Forest classifier represented by the curve, it is smoother. XGBoost model also exhibits similar strong performance in terms of its macro-average AUC value, which is close to 0.99 as shown in (Figure 5). In the ROC curve, the line closely hugs the upper-left border. This shows that the ensemble model effectively handles class variation by maintaining higher predictive accuracy. CatBoost results show near perfect classification since the AUC value for classification itself is 1.00 as shown in (Figure 6). Further, the ROC curve for the particular class of letter classification is close to a right angle with the top left part of the graph, which means Cat Boost has made this particular class of classification with utmost accuracy. Among all models, CNN is the one that distinguishes itself the most. As shown in the graph for the lower- and upper-case letter models, the AUC values are almost fixed around 0.998–0.999 as shown in (Figure 7). The curves appear to converge toward the upper-left corner of the plot. It means the CNN model has the highest capacity when it comes to classification of visual patterns and figures of handwritten characters.

Sl.No	Classifier	Accuracy	Precision	Recall
1	SVM	88%	85%	87%
2	Random Forest	91%	89%	90%
3	XG Boost Ensemble	96.87%	95%	96%
4	Cat Boost	91%	89%	91%
5	k-NN	90%	88%	88%
6	CNN	98%	98%	97%

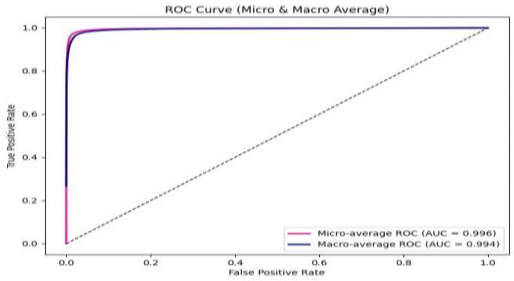

Figure 3. ROC Curves Using SVM

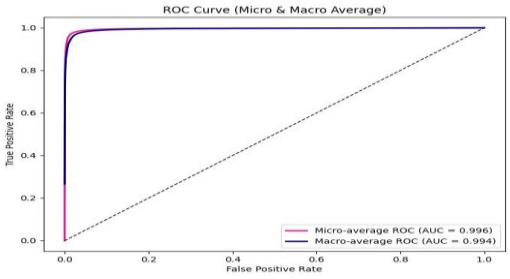

Figure 4. ROC Curve Using Random forest

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

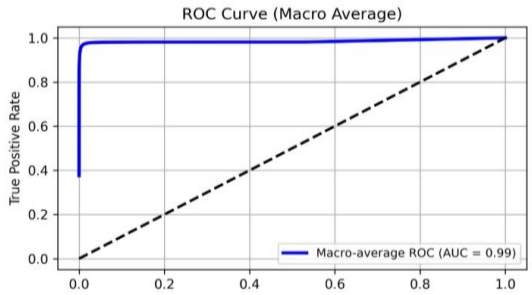

Figure 5. ROC Curve Using XGBoost Ensemble

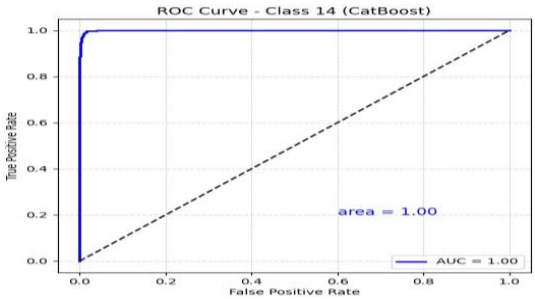

Figure 6. ROC Curve Using Cat Boost

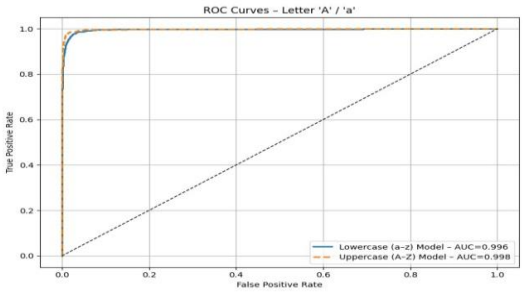

Figure 7. ROC Curve Using CNN

4.1.2 Confusion Matrix

The confusion matrices also indicate the performance of the model for handwritten characters in both lowercase and uppercase. The lowercase matrix has a lot of correct predictions across the diagonal, though some similar shapes occasionally get mixed up. The uppercase matrix has instead a much sharper diagonal, the model being more confident in its predictions on uppercase letters and making fewer mistakes. From the confusion matrices, it is also possible to note the level of accuracy of the model concerning handwritten characters. Considering the model for lowercase characters as represented in (Figure 8), it is possible to see that the model has been able to make accurate predictions along the diagonal. There are a number of cases where different characters appear in similar shapes, and as a result, characters like c, o, v, and u overlap. Despite these rare cases of confusion between certain characters, it is evident that the model makes accurate predictions by placing most of the characters in the expected location, which indicates that it understands lowercase characters well. On the other hand, (Figure 9) indicating the confusion matrix for the uppercase model hints that it makes sharper and clearer predictions compared to (Figure 8). This is because the shapes of uppercase characters make it easy for the model to classify each character as represented along the diagonal. On average for both cases the model count be counted upon to respond well, it's performance seems very slightly more repeatable with upper case but not a lot.

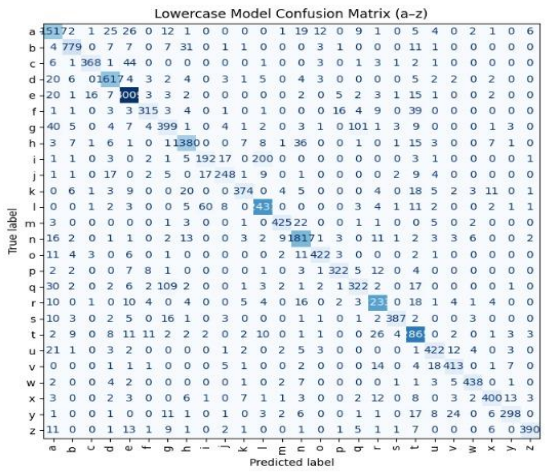

Figure 8. Confusion Matrix for Lowercase Model

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

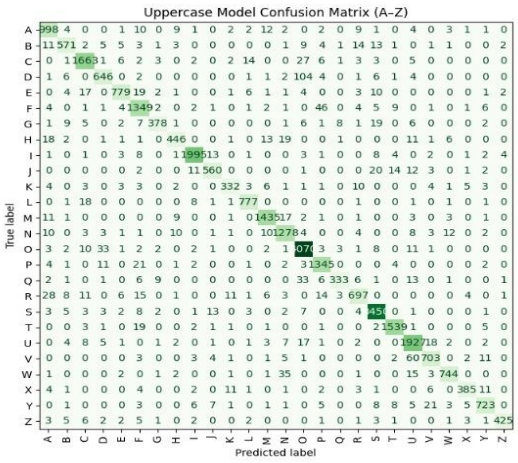

Figure 9. Confusion Matrix for Uppercase Model

4.1.3 Model Predicted Output

Some sample predictions of the model on the EMNIST dataset and some real handwritten characters are shown in the (Figure 10) and (Figure 11). Some clean handwriting letters from the EMNIST dataset and the predicted outputs from our model as shown in (Figure 10). In real life, the system can learn various styles of handwriting from the dataset. To ensure that the model really works for real handwriting, we involved actual handwritten samples from students. These real samples are shown in (Figure 11), where each letter was written on paper, then photographed and processed through our model to see its prediction. These examples show how the model will perform, not only on the standardized pictures of the dataset but also on real-life handwritten characters, proving to be capable of dealing with different styles of handwriting in real life.

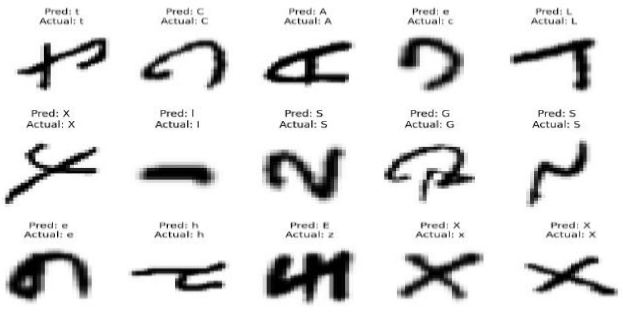

Figure 10. Sample Predictions on EMNIST Dataset

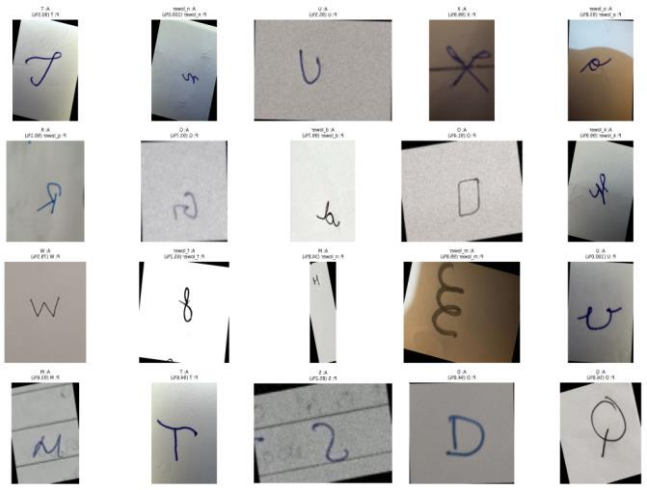

Figure 11. Real Handwritten Characters Collected from Students and Model Predictions

4.2 DISCUSSION

Handwritten recognition in the world of Artificial Intelligience and Machine Learning has been associated with many obstacles in recent years. One of the major problems is quality of images, because OCR systems need to process various size images, different light condition and noise level and contrast etc. especially in case of handwritten recognition [10]. From this outcome, we found that for handwritten character recognition various accuracy were acquired with different models of machine learning. One of the single machine learning best- performing classifiers was XGBoost which yielded an accuracy of 96.87%, indicating that it could discern complex patterns more compared to other single classifiers.

The XGBoost Ensemble model also demonstrated good performance, because the addition of several learners improved stability of predictions over using a single learner. Pre-processing methods including image conversion into grayscale, noise elimination, and resizing of the images and data augmentation, were vital to help increase overall

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

accuracy. These techniques made the models better at managing messy handwriting, inconsistent lighting and differences in writing styles.

We also experimented with a deep learning based CNN model in addition to our machine learning models. The CNN model performed better with an accuracy of 98% as it automatically learns relevant features from the images itself and does not need feature selection to be done manually. Its ROC curve was also very close to the top-left corner, suggesting highly accurate classification. This demonstrates that CNN is more appropriate for isolated handwritten character recognition task compared to the other cases and for larger databases and complex handwriting recording.

In conclusion, our results reveal that classical machine learning models can do well in OCR given appropriate training with good preprocessing steps, nonetheless the deep learning model config CNN came out the best of all machine learning classifiers. Thus CNN is a useful and effective technique for the accurate recognition of handwritten characters. Further gains can also be expected by employing larger data sets, through the use of deeper CNN architectures or advanced techniques like dropout tuning and batch normalization.

REFERENCES

[1] Dr. V. Geetha, Ch V V Sudheer, A V Saikumar , Dr C K Gomathy . 2022 March 2022, Journal of Engineering, Computing & Architecture (Volume 12, Issue 3, ISSN: 1934-7197). “Optical Character Recognition”

[2] Ameer Majeed, Hossein Hassani. 24 Aug 2024, “Ancient but Digitized: Developing Handwritten Optical Character Recognition for East Syriac Script Through Creating KHAMIS Dataset”

[3] Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, Furu Wei. September 6 2022, Beihang University, Microsoft Corporation “TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models”

[4] Syed Sajid Ullah, Li Gang, Mudassir Riaz, Ahsan Ashfaq, Salman Khan, Sajawal Khan. March 2025, “Handwritten Digit Recognition: An Ensemble-Based Approach for Superior Performance”

[5] Avinash Malladhi. April 30 2023, International Journal of Computer Trends and Technology “Transforming Information Extraction: AI and Machine Learning in Optical Character Recognition Systems and Applications Across Industries”

[6] Dibya Thapa and Rebika Rai. 2 October 2025, “FREQ-EER: A Novel Frequency-Driven Ensemble Framework for Emotion Recognition and Classification of EEG Signals”

[7] Jamshed Memon ,Mairasami ,Rizwan Ahmedkhan. July 28 2020, “Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)”

[8] Mahmoud SalahEldin Kasem (2023) “Advancements and Challenges in Arabic Optical Character Recognition: A Comprehensive Survey”

[9] Yash Agrawal (2024) “Optical Character Recognition for Ashokan Brahmi Inscriptions”

[10] Senka Drobac and Krister Linden (2020) “Optical Character Recognition with Neural Networks and Post-Correction with Finite State Methods”

[11] Rohini G. Khalkar, Adarsh Singh Dikhit, Anirudh Goel, Manisha Gupta. International Advanced Research Journal in Science, Engineering and Technology Vol. 8, Issue 6, June 2021. “Handwritten Text Recognition using Deep Learning (CNN & RNN)”

[12] HusamAhmadAlhamad1 ,MohammadShehab1, MohdKhaled Y. Shambour, MuhannadA.Abu-Hashem3 2024 “Handwritten Recognition Techniques: A Comprehensive Review”

[13] Yitao Yao 2024 “Analysis for Advancements of Optical Character Recognition in Handwriting Recognition”

[14] Nahar, K. M., Alsmadi, I., Al Mamlook, R. E., Nasayreh, A., Gharaibeh, H., Almuflih, A. S., & Alasim, F. (2023) “Recognition of Arabic air-written letters: machine learning, convolutional neural networks, and optical character recognition (OCR) techniques”

[15] Srivastava, S., Verma, A., & Sharma, S. (2022, February). “Optical character recognition techniques: A review” IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS) (pp. 1-6). IEEE.

[16] Devi, S. N., & Fatima, N. S. (2025). “Handwritten optical character recognition using TransRNN trained with self improved flower pollination algorithm (SI-FPA)”. Multimedia Tools and Applications, 84(18), 19947-19969.

[17] Raza, S. A., Farooq, M. S., Farooq, U., Karamti, H., Khurshaid, T., & Ashraf, I. (2025). “A Convolutional Neural Network Based Optical Character Recognition for Purely Handwritten Characters and Digits”.

[18] Alejandro Baldominos, Yago Saez, Pedro Isasi.. August 4, 2019, Applied Sciences Journal “A Survey of Handwritten Character Recognition with MNIST and EMNIST”

[19] N Arica, FT Yarman-Vural IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 24, Issue: 6, June 2002) “Optical character recognition for cursive handwriting”

[20] Priyanka Kaushik, Priyanka Rawat, Devyansh Batra, P Vensheeda Delin, S Kaliappan. (2025) “Python- Based Optical character Recognition (OCR)”