International Journal on Smart & Sustainable Intelligent Computing
Vol. 03, Iss. 01, S. No. 03, pp. 25–50, March 2026
ISSN (Online): 3048-8508

Received: 06 Feb 2026, Accepted: 23 Feb 2026, Published: 20 April 2026
DOI: https://doi.org/10.63503/j.ijssic.2026.239

**Research Article**

# A CNN Framework for Real-Time and Edge Deployment with High Accuracy for Lightweight Deep Learning for Bengali OCR

Dipankar Dey[1], Dipak Kumar Jana[2]
[1] Global Institute of Science and Technology, Haldia, WB, India
[2] Gangarampur College, Gangarampur, Dakshin Dinajpur-733124, WB, India
Email: deydipankar2014@gmail.com, dipakjana@gmail.com
*Corresponding Author: Dipankar Dey, deydipankar2014@gmail.com

## Abstract

The Bengali script, like other Indic scripts, has a large character set, complex conjunct characters, and varying handwriting styles, making optical character recognition (OCR) a complex task. Although convolutional neural networks (CNNs), ensemble techniques, and CNN-Transformer models have shown high accuracy, their high computational complexity and large input size make them unsuitable for mobile and embedded systems. To make the processing cost-effective and improve the accuracy of handwritten Bengali character recognition, this paper proposes a light-weight CNN with optimized filters and a $40 \times 40$ input size. Compared to the existing deep learning and hybrid approaches, the proposed approach has shown 98.29% top-1 accuracy on a 50-class dataset with 12,000 training samples and 2,997 testing samples, reducing the parameter size and computational complexity. A brief literature review is also presented to discuss the evolution of real-time and large-scale Bengali OCR systems and the challenges involved in compound character recognition, efficiency, and system-level validation.

**Keywords:** Bengali OCR, Low-Resource Script Processing, End-to-End Deep Models, Real-Time Handwritten Character Recognition, Lightweight CNN Architecture

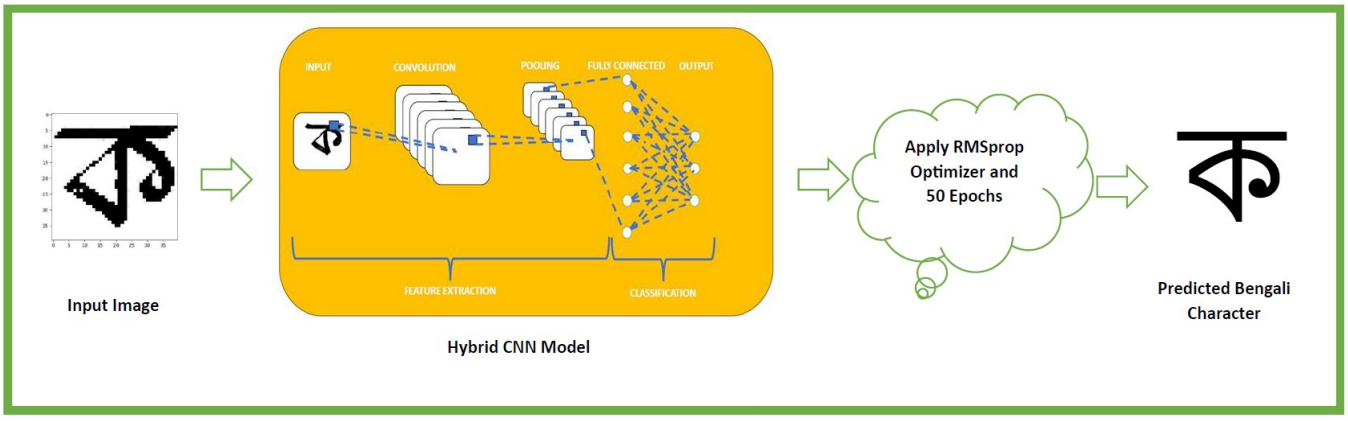



Figure 1: Flow chart of the proposed model

## 1 Introduction

Due to the popularity of Bengali as a means of communication and being one of the major languages of India, accurate character recognition is required for effective digital processing. Although the recognition of handwritten Bengali has made substantial progress, higher accuracy and feasibility are still required. Most of the previous

works on handwritten Bengali recognition have used conventional image processing and computational intelligence approaches, which are heavily dependent on human-designed algorithms and manual feature extraction. However, for advanced visual pattern recognition, recent breakthroughs in deep learning, specifically CNN ("Convolutional Neural Networks"), provide a relatively more autonomous and accurate solution. This has improved its utility in computer vision for recognizing handwritten numbers and characters. These technologies are critical for Optical Character Recognition (OCR) applications, which in turn promote further research in Bengali and other popular languages.

It is anticipated that the proposed framework for handwritten Bengali character prediction would speed up the processes of character classification and image collection. The CNN-based classification, feature extraction, and preprocessing are perfectly combined in this process. The workflow of the proposed scheme is shown in Figure 1.

In the application of OCR ("Optical Character Recognition") technology for the translation of printed and handwritten texts into machine-readable form, the accurate recognition of Bengali text is a very important step [2]. This process has been used in different sectors to facilitate data entry, document processing, and information extraction. Additionally, it helps in the very important NLP ("Natural Language Processing") processes of sentiment analysis, automated summarization, and translation. However, the decoding of the Bengali script is hard due to the large number of characters and the different writing styles. The proposed scheme addresses different challenges by demonstrating a practical CNN-based approach for character classification. By adjusting the recognition accuracy to suit the needs of Bengali consumers, the proposed solution is expected to enhance OCR and language recognition technology.

The prediction model of the Bengali script, as discussed in this paper, has some limitations due to its reliance on manual feature engineering and traditional algorithms. The proposed algorithm uses the capability of convolutional neural networks to overcome various obstacles and relies on a visual Keras platform to demonstrate the structure of the algorithm. After conducting extensive experimentation, this scheme has been able to achieve a validation accuracy of 0.9643 (96.43%) and a training accuracy of 0.9829 (98.29%), which is a remarkable achievement. The proposed algorithm is able to prove the effectiveness of CNN-based learning in accurately identifying Bengali characters. Apart from its theoretical significance, the proposed algorithm is also useful in real-life applications related to language processing and optical character recognition, which helps to improve text recognition in the digital world. Despite its limitations, the proposed algorithm is able to achieve a high accuracy rate of 98.29%, which proves the effectiveness of CNN models in overcoming various obstacles to achieve accurate Bengali character recognition.

This research work presents a lightweight CNN-based system for trustworthy Bengali character recognition, which handles a vast vowel and consonant alphabet as well as handwriting variations. The proposed system uses automated deep learning for feature extraction, which improves efficiency and scalability without the need for manual feature extraction. The result of the experiment shows excellent performance with a training accuracy of 0.9829 (98.29%) and a validation accuracy of 0.9643 (96.43%), which surpasses conventional methods. The system facilitates the digitization of handwritten and printed text, thereby speeding up data entry, document processing, and information retrieval. In addition, it facilitates summarization, sentiment analysis, and translation processes. The proposed $40 \times 40$ CNN-based system is more suitable for real-time Bengali OCR on resource-constrained devices because it provides better accuracy with reduced complexity compared to resource-hungry hybrid/ensemble methods.

## 1.1 Research Contributions

- Suggests a small CNN-based system for handwritten Bengali character recognition. It manages a wide range of characters and handwriting variations.
- Achieves high performance with 98.29% training accuracy and 96.43% validation accuracy. It also reduces parameters and inference time.
- Enables efficient digitization, document processing, and rapid information retrieval from Bengali text, all of which enhance OCR capabilities.
- Supports NLP activities that come after, such as summarization, sentiment analysis, and translation. Bengali language technologies are enhanced by this.

- Identifies research gaps and practical difficulties in Bengali and related Indic OCR by conducting a systematic survey.
- Delivers a lightweight, resource-efficient design suitable for real-time deployment on mobile and embedded platforms.

### 1.2 Summary of Figures and Tables

All of the figures and tables used in this work, together with their titles and purposes, are summarized in Table 1. It offers a fast way to access the tabular and graphic components of the suggested Bengali OCR system.

### 1.3 Road map of the proposed scheme

The proposed paper has been designed as follows: Section 2 gives an overview of the traditional models on Bengali script recognition and CNN-based solutions. Section 3 explains the proposed solution, including the preparation of the dataset, preprocessing, architecture, and training for effective character prediction. Section 4 describe the experimental findings, including training and validation performance, testing accuracy, confusion matrices, and other measures of precision, F1-score, and recall, compared to the existing techniques. Section 5 provides a summary of the key results, while Section 6 outlines potential future directions, including transfer learning, federated learning, and real-time systems. Finally, Section 7 describes our existing scheme limitations and potential enhancements.

## 2 Previous Studies

Riffat Sharmin et al. use a CNN to classify spoken Bengali numbers. Although strongest in signal processing, their work remains exclusively audio-centric and ignores image-based optical character recognition.

Xingyue Sun et al. [2] had discuss a CNN model to estimate material fatigue life under multiaxial low-cycle loading conditions. Md. Saif Hassan Onim et al. introduce BLPnet, a deep learning solution developed exclusively for Bengali license plate recognition. This solution is highly specialized and does not address script recognition problems, although it is highly accurate in its application area. It performs very well on printed license plate text but poorly on handwritten Bengali numerals.

For the classification of Bengali handwriting digits, Abu Sufian et al. [4] proposed BDNet, a densely connected CNN. It concentrates only on digits and doesn't include the entire set of Bengali characters, including compound characters, even though it achieves high accuracy on digits.

Md. Rajib Hossain et al. [5] used a deep CNN for the classification of Bengali text documents. It is very good for document-level classification, but it doesn't provide enough detail for character-level OCR.

Rajib Ghosh et al. [6] proposed an RNN-based solution for handwritten text identification in Bengali and Devanagari scripts. It is accurate but not suitable for isolated character recognition, and it has a high computational overhead.

A CNN-BiLSTM hybrid was used by Himanish Shekhar Das et al. [7] to identify Indian languages from spectrogram pictures. While it works well for language detection, individual character OCR jobs cannot directly use it.

In order to recognize Bangladeshi Sign Language, Md. Monirul Islam et al. [8] applied transfer learning model. The method does not work with handwritten or printed Bengali text; it is specifically designed for gesture recognition.

For handwritten digit recognition, Savita Ahlawat et al. [9] suggested a hybrid CNN-SVM system. Although it is effective at identifying numbers, it ignores the intricacy of full-script OCR.

For the recognition of Bangla sign language, Sunanda Das et al. [10] created a hybrid deep transfer learning technique that combines random forest classification. This approach is not intended for OCR applications and is domain-specific.

BornoNet, a CNN-driven classifier of handwritten characters in Bangla, was introduced by Akm Shahariar Azad Rabby et al. [11]. However, scalability and resilience were limited because the assessment was conducted on a small dataset.

Table 1: Summary of Figures and Tables used in this paper with their references and purposes.

| Type | Caption / Reference | Purpose |
|---|---|---|
| Figure 1 | Graphical Abstract | Provides an overview of the proposed lightweight CNN framework for Bengali character recognition. |
| Figure 2 | Handwritten Bengali Characters Sample | Displays sample input images from the dataset used for training and testing. |
| Figure 3 | Diagrammatic representation of a fundamental CNN framework | Explains the basic structure and flow of a CNN model for OCR tasks. |
| Figure 4 | Illustration depicting the structure of a fundamental CNN model | Presents a detailed schematic of a CNN architecture used for baseline comparison. |
| Figure 5 | Architecture of proposed CNN model | Shows the detailed architecture of the proposed CNN layers and structure. |
| Figure 6 | Graphical Representation of proposed CNN model | Visualkeras-based visualization of the designed CNN for Bengali character recognition. |
| Figure 7 | Experimental outcomes of different predicted images | Demonstrates model predictions on unseen handwritten Bengali characters. |
| Figure 8 | Graphical representation of Epochs and Training Loss Analysis | Plots training loss vs epochs to show convergence and optimization. |
| Figure 9 | Graphical representation of Epochs and Training/Validation Accuracy comparison | Compares training and validation accuracy trends across epochs. |
| Figure 10 | Confusion matrix of proposed Model | Visual summary of classification performance across 50 Bengali character classes. |
| Figure 12 | Accuracy comparison of proposed model with existing OCR approaches | Shows comparative accuracy of proposed CNN against other state-of-the-art models. |
| Figure 11 | Accuracy and loss gap analysis between training and validation across epochs | Illustrates the gap trends between training and validation metrics to highlight generalization performance. |
| Table 2 | Summary of Literature Review: Models and Identified Research Gaps | Summarizes prior works, their models, and gaps in Bengali OCR research. |
| Table 3 | Comparative Analysis of Bengali Script and Related Recognition Studies | Provides dataset sizes, classes, input dimensions, and accuracy of related studies. |
| Table 4 | Taxonomy of Bengali OCR Works Based on Task and Methodology | Categorizes prior works into digit recognition, isolated character, word/document OCR, and hybrid models. |
| Table 5 | Model architect summary | Lists the detailed CNN layers, output shapes, and parameter counts of the proposed model. |
| Table 6 | Test Accuracy of Proposed Model | Shows epoch-wise training/validation loss and accuracy values. |
| Table 7 | Statistical Evaluation of the Proposed Model | Reports precision, recall, F1-score, and support for the dataset. |
| Table 8 | Ablation Study on the Proposed Bengali Character Recognition Model | Analyzes impact of input size, dropout, and filter configuration on accuracy. |
| Table 9 | Error Analysis based on Accuracy Gap and Loss Gap across Epochs | Summarizes class-wise error trends and highlights challenging cases for the model. |
| Table 10 | Statistical Evaluation of the Proposed Model (Comparison) | Compares proposed work's accuracy with other OCR approaches. |

M. Antony Robert Raj et al. [12] concentrated on the application of statistical methods for Tamil handwritten text understanding. The process's cross-language adaptability has not been confirmed because it has not been tested on Bengali scripts.

CNN transfer learning was utilized by Sandeep Dwarkanath Pande et al. [13] to digitize Devanagari texts. Although it works well for Devanagari, it hasn't been modified to recognize Bengali script.

To recognize Bangla letters and numbers, Sourajit Saha et al. [14] created a customized deep neural network. Although it is ideal for high precision, it comes with a large, computationally demanding model that is challenging to apply on devices with limited resources.

N. Majid et al. [15] developed an offline system for Bangla character recognition. The technique is strongly linked to their dataset and has poor generalization capabilities for different writing styles.

Muhammad Aminur Rahaman et al. [16] proposed a six-layer CNN architecture for sign language recognition of numerals and alphabets. This gesture-based model is irrelevant to OCR tasks.

Hoque et al. [17] proposed a Transformer-ensemble model for Bengali sentiment analysis. They tested their model on 11,807 comments with five different pre-trained models (mBERT, BanglaBERT, Bangla-Bert-Base, DistilmBERT, and XLM-R-base). Their model performed exceptionally well, with 95.97% accuracy and 95.96% F1-score, outperforming state-of-the-art Bengali SA models.

Based on the survey of the literature, three prominent patterns have been identified: first, traditional machine learning techniques are not scalable; second, CNNs are strong baselines but often require high-resolution inputs and high memory; and third, hybrid CNN-Transformer configurations often provide strong baselines but at the expense of computational efficiency. Despite these findings, there is no existing work that has demonstrated a lightweight CNN outperforming the hybrid configurations and still being able to support real-time processing. This paper directly addresses this problem.

**Summary of Research Gaps:** Previous research frequently uses computationally demanding models, focuses on small tasks like numbers or single phrases, and covers only a portion of the Bengali character set. Although they rely on deep, resource-demanding networks, techniques like BornoNet [16] and AKHCRNet [14] achieve good accuracy. This paper suggests a lightweight CNN that minimizes complexity while retaining good accuracy for offline handwriting recognition in order to overcome these problems. Additionally, the majority of the research that has already been done is domain-specific; therefore, this study provides a unified comparative analysis across categories in order to present a more comprehensive and useful viewpoint.

In contrast to the conventional Bengali OCR system that relied on hand-engineered features or a hybrid approach, the proposed system relies entirely on an end-to-end CNN architecture that requires no human intervention in feature design. As pointed out in Table 2, the existing literature faces challenges related to dataset dependency, poor adaptability to varying handwriting styles, and high computational complexity. The proposed $40 \times 40$ CNN architecture reduces complexity by integrating feature design and classification into a single, seamless process, thus increasing the efficiency and originality of the system.

## 2.1 Comparative Analysis of Existing Works

Bengali handwriting digit recognition, character classification, text categorization, and sign language recognition have all been documented in a number of studies. Although each has made a substantial contribution, input representation, class variety, and dataset type all have a major impact on how well each performs. Table 3 presents 10 pertinent research in terms of dataset size, number of classes, input dimension, and recognition accuracy to give a succinct comparison.

It is clear from Table 3 that current approaches cover a variety of Bengali recognition tasks, including text categorization [5], isolated characters [11], digits [4, 9], and sign language recognition [8, 10]. The originality and purpose of our suggested strategy are highlighted by the fact that relatively few studies specifically target offline Bengali script recognition with a lightweight CNN model.

Based on the literature, the current best practices are categorized into four groups: hybrid deep learning models for scene text OCR, isolated character recognition, word recognition, and digit recognition. Although these methods show robust identification performance, they are still scattered and may only focus on a limited set of Bengali characters or require a significant amount of computational resources. This paper will give a survey-based outlook and serve as a benchmark for the proposed lightweight CNN architecture.

Table 2: Summary of Literature Review: Models and Identified Research Gaps

| Author(s) | Model / Approach | Research Gap Identified |
|---|---|---|
| Riffat Sharmin et al. [1] | CNN for Bengali spoken digit classification | Audio-focused; lacks image-based OCR capability. |
| Xingyue Sun et al. [2] | CNN for material fatigue prediction | Domain-specific; not suited for script recognition. |
| Md. Saif Hassan Onim et al. [3] | BLPnet for license plate OCR | Limited to license plates; poor generalization to handwriting. |
| Abu Sufian et al. [4] | BDNet for Bengali numerals | Works only for numerals; ignores full script complexity. |
| Md. Rajib Hossain et al. [5] | Deep CNN for document categorization | Document-level only; lacks character-level detail. |
| Rajib Ghosh et al. [6] | RNN for handwritten word recognition | High computational cost; not optimal for isolated characters. |
| Himanish Shekhar Das et al. [7] | CNN-BiLSTM for language identification | Language-level focus; not tailored for OCR tasks. |
| Md. Monirul Islam et al. [8] | Transfer learning for sign language | Gesture-specific; unrelated to text OCR. |
| Savita Ahlawat et al. [9] | Hybrid CNN-SVM for digit recognition | Digit-only focus; not for complex characters. |
| Sunanda Das et al. [10] | Hybrid deep transfer learning for sign language | Sign recognition only; no OCR application. |
| Akm Shahariar Azad Rabby et al. [11] | BornoNet CNN for Bangla handwriting | Small dataset; limited scalability. |
| M. Antony Robert Raj et al. [12] | Tamil character recognition | Tested on Tamil only; unverified for Bengali script. |
| Sandeep Dwarkanath Pande et al. [13] | CNN for Devanagari text | Domain-specific; no Bengali adaptation. |
| Sourajit Saha et al. [14] | Custom DNN for Bangla characters | High accuracy but resource-heavy. |
| N. Majid et al. [15] | Offline Bangla character recognizer | Dataset-specific; limited generalization. |
| Muhammad Aminur Rahaman et al. [16] | Six-layer CNN for sign language | Gesture-focused; not for standard OCR. |
| Md. Nesarul Hoque et al. [17] | Transformer-ensemble (mBERT, BanglaBERT, DistilmBERT, XLM-RoBERTa) for Bengali sentiment analysis | Text-based only; no OCR or visual script recognition capability. |

Table 3: Comparative Analysis of Bengali Script and Related Recognition Studies

| Study | Dataset Size | Classes | Input Dim | Accuracy |
|---|---|---|---|---|
| Sharmin et al. [1] | 3,000 spoken digit samples | 10 | Audio spectrograms | 98.5% |
| Sufian et al. (BD-Net) [4] | 60,000 numeral images | 10 | 28×28 | 98–99% |
| Hossain et al. [5] | 156,207 docs (BDTC) | 13 | 416×416 tokens | 96.96% |
| Ghosh et al. [6] | Online stroke dataset | 100+ words | Variable-length | 95.24% |
| Himanish et al. [7] | Multilingual dataset | 7 languages | Variable-length audio | 93.2% |
| Monirul et al. [8] | Static sign language images | 39 | 224×224 | 97.8% |
| Savita et al. [9] | MNIST (60,000) | 10 digits | 28×28 | 99.2% |
| Das et al. [10] | 8,000 sign language images | 50 | 224×224 | 98.1% |
| Rabby et al. [11] | BanglaLekha (166K) | 84 characters | 32×32 grayscale | 96.8% |

Currently, the research on Bengali optical character recognition (OCR) has progressed along four major branches: hybrid systems that combine deep learning with scene text recognition, word recognition, isolated character recognition, and digit recognition. Although there has been some level of advancement in each of these branches, most of the existing work is still fragmented, meaning that they might focus only on a certain set of characters or require intensive computation. This paper provides a survey perspective and can be used as a benchmark for comparison with the proposed lightweight CNN approach, which combines all four branches. The CNN-based approach represents an advancement over traditional OCR systems because they have progressed beyond the need for hand-engineered features to hybrid models such as CNN-RNN and Transformers. The convolutional neural networks increase the robustness of the system by learning hierarchical features from raw data, which are less dependent on strict geometric or statistical models. The positioning of our contribution within this rapidly developing area emphasizes the efficiency of the lightweight CNN and its relevance to the overall development in the OCR algorithm area.

Table 4: Taxonomy of Bengali OCR Works Based on Task and Methodology

| Category | Representative Works | Key Focus / Limitation |
|---|---|---|
| Digit Recognition | [1], [4], [9], [14] | Focused on isolated Bengali digits; achieved strong accuracy but limited to numerals only. |
| Isolated Character Recognition | [11], [15], [14] | Problem of Bengali characters; struggled with compound characters and handwriting variability. |
| Word-Level / Document OCR | [5], [6] | Large word recognition or full text documents; challenges with segmentation, noise, and diverse writing styles. |
| Hybrid / Deep Learning Approaches | [3], [7], [10] | Applied CNN–BiLSTM, CNN–Transformer, or hybrid architectures; increase accuracy but required large input sizes or heavy computation. |

The taxonomy of Bengali OCR works that we have discussed is presented in Table 4, which categorizes the works into four categories: digit recognition, isolated character recognition, word-level OCR, and hybrid deep learning techniques. The taxonomy reflects the evolution of works from simple digit and character recognition to more complex document-level recognition tasks and hybrid models, while also resonating with the typical constraints that motivate the design of the lightweight CNN in this work.

# 3 Methodology

Looking at the bigger picture of how the system is being constructed, piece by piece, and function by function, our CNN is unique and very effective at predicting Bengali handwritten characters. The dataset contains 12,000 grayscale training images and 2,997 test images, all of which are normalized to 40x40 pixels. Image preprocessing techniques are used to enhance the images. The prediction model is a CNN that is famous for its character recognition abilities and is represented graphically as a Keras model.

Although this section is focused on the proposed CNN, it is important to point out that the research survey on Indic OCR includes a variety of techniques, including CNN-based recognition techniques. Although new hybrid models such as CNN-RNN and CNN-Transformer have emerged to better model sequences, most of the older models were designed by hand and relied very heavily on feature engineering. Looking at the bigger picture, our design fits in with these older trends.

The emergence of CNN-based approaches represents a turning point for hybrid CNN models in OCR, going beyond the traditional, manually designed concepts of the past. In contrast to traditional shape-based and statistical approaches, which are based on hand-designed features, CNNs have the capability to learn features autonomously, and when combined with RNNs or Transformers, they are particularly effective at processing sequential data. Our research is part of this line of research, emphasizing two important aspects: the efficiency of our lightweight CNN approach and its increasing relevance to the OCR community.

Our lightweight CNN approach is well-integrated into the current OCR state-of-the-art. In the early days of Bengali OCR, traditional statistical classifiers and manually designed features dominated, but CNN-based models quickly gained popularity because of their capacity to learn layered representations autonomously. With the increasing maturity of the field, models based on Transformers and hybrid CNN-RNN models gained popularity for their improved sequence processing capabilities. This work demonstrates that a well-designed small CNN can actually perform better than these models while consuming less computational resources.

## 3.1 Dataset

This work uses a Bengali handwritten character dataset, which you can download from GitHub at https://github.com/mehedihasanbijoy/Bangla-Handwritten-Character-Recognition. The dataset has been preprocessed and split into training and testing sets, where each image is a 40x40 pixel grayscale image. The prepared dataset is used to evaluate the ability of the proposed model to predict the handwritten characters. Information about the implementation, including software tools, algorithms, and hardware, will be explained in the following sections. The main dataset consists of 50 classes, with 12,000 training samples and 2,997 test samples. In a broader context, the work also takes into consideration other datasets, such as BanglaLekha-Isolated (with around 166,000 images in 84 classes) and CMATERdb. To improve generalization and avoid overfitting to specific writers, writer-independent splits are used to split training and testing sets. Figure (2) shows the sample characters.

## 3.2 Training Details

The proposed approach trains a CNN on the task of recognizing handwritten Bengali characters. The network is designed to achieve a balance between accuracy and efficiency, and the training process is done using the latest techniques to ensure that the network performs well. The hyperparameters of the network are set using experiments and insights that are data-driven and come from sound reasoning. By experimenting with the hyperparameters and observing how well the network performs on a validation set, the authors are able to determine the best combination that improves the generalization and robustness of the network. This allows the network to achieve a training accuracy of 0.9829 (98.29%) and a validation accuracy of 0.9643 (96.43%).

### 3.2.1 Filters

The number of filters is one of the key points that increase the complexity of the convolutional layers. To help the network learn complex features, the first Conv2D layer has 32 filters, the second Conv2D layer has 64 filters, and the third Conv2D layer has 96 filters.


Figure 2: Handwritten Bengali Characters Sample

### 3.2.2 Kernel Size

With a kernel size of $(3, 3)$, filters may explore the data for fine features, enabling the model to extract fine information from the input images.

- Activation Function: The activation function relu ("Rectified Linear Unit") brings non-linearity to the operations of our propose scheme, making it possible to extract complex patterns from the data.
- Input Shape: This scheme is structured to control grayscale images of size $40 \times 40$ pixels, which are the input sizes necessary for character recognition.
- Pooling: MaxPooling2D layers with a (2, 2) pooling window are applied to systematically downscale the input data, thus reducing the spatial dimensions and improving the extraction of prominent features.
- Dropout: The Flatten layer is applied to convert the result of the previous layers into a 1-D vector, which is then ready for the fully connected layers.
- Fully Connected Layers: The model comprises two Dense layers with 512 units and 50 units, respectively. The first Dense layer uses a ReLU activation function, which helps in feature representation, while the second Dense layer uses a softmax activation function for character classification.
- Optimizer: This scheme is structured with the RMSprop optimizer with particular parameters, which include a learning rate of 0.001, rho of 0.9, epsilon of 1e-08, and decay of 0.0.
- Loss Function: The loss function used in this case is the categorical cross-entropy loss function, which calculates the difference between the predicted and actual class labels in the multi-class classification problem.
- Metrics: The accuracy metric is used in the training phase to gauge the model's performance. The number of successfully categorized classes is measured by this statistic.

Table 5: Model architect summary

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d (Conv2D) | (None, 40, 40, 32) | 896 |
| max_pooling2d (MaxPooling 2D) | (None, 20, 20, 32) | 0 |
| conv2d_1 (Conv2D) | (None, 20, 20, 64) | 18496 |
| max_pooling2d_1 (MaxPooling 2D) | (None, 10, 10, 64) | 0 |
| conv2d_2 (Conv2D) | (None, 10, 10, 96) | 55392 |
| max_pooling2d_2 (MaxPooling 2D) | (None, 5, 5, 96) | 0 |
| flatten (Flatten) | (None, 2400) | 0 |
| dense (Dense) | (None, 512) | 1229312 |
| dropout (Dropout) | (None, 512) | 0 |
| dense_1 (Dense) | (None, 50) | 25650 |

Total params: 1,329,746
Trainable params: 1,329,746
Non-trainable params: 0

After 50 epochs of training, the learning ability of our approach is very strong. The training accuracy reaches 0.9929, which shows excellent ability to identify complex patterns. The validation accuracy of 0.9643 shows excellent generalization capabilities for new data, and a very small loss of 0.0038 indicates overall effectiveness for Bengali character recognition tasks.

The excellent performance for predicting handwritten Bengali characters is achieved through optimal hyperparameter adjustment. Depending on the understanding and experiments conducted in the previous models, the authors have adjusted the hyperparameters of the model to improve performance. The strategy achieves a good balance between accuracy, complexity, and cost. By taking advantage of the capability of CNN to extract high-level features, it provides accurate predictions. The experimental results demonstrate the efficiency of the approach, with a test accuracy of 0.9643 and a test loss of 0.0038. Our scheme highlights the effectiveness of the approach for Bengali character recognition and its applicability for optical character recognition and language processing. The architecture of the model is presented in Table 5, which illustrate the application of CNNs for predicting handwritten Bengali characters.

A very useful tool for comprehending our work is the model summary. This provides the architecture, hyperparameters, layers, output formats, and the total number of parameters. The CNN architecture is broken down layer by layer in Figure 3, which also explains the reason for each step. A very detailed description is required to comprehend how the model works and why it is useful. To comprehend how the model works to predict Bengali characters, you must understand the layer by layer architecture of the model. Our aim is to develop a system that can predict Bengali characters from handwritten images, which is achieved by carefully selecting and adjusting the hyperparameters of the deep learning model, including logistic regression.

Figure 3 illustrates the proposed model's layered structure, which contains CNN model definitions. The model's efficacy is increased by each layer, each of which plays a distinct role. The model's data processing and flow are made clearer by the architecture. The architecture contains important details regarding the model's activation functions, the number and kind of filters in the convolutional layers, and the size of the input. To understand how the model processes incoming data and extracts characteristics, it is essential to comprehend this tiered structure. The complete architecture of the suggested model is shown in Figure 3, which provides lucid insights into its construction and aids in assessing its performance and accuracy in Bengali character prediction.

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

Equation 1 represents the categorical cross-entropy loss function. It includes the true label $y_i$ and the predicted

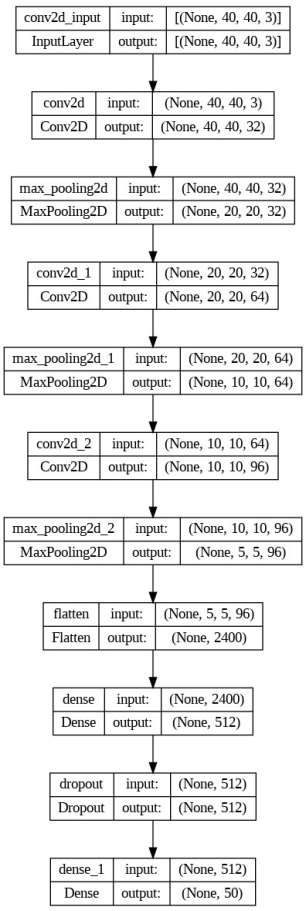



Figure 3: Diagrammatic representation of a fundamental CNN framework

probability $\hat{y}_i$. The training process optimizes the loss function for classifying Bengali characters.

### 3.3 CNN Model Design

The CNN is set up to recognize the handwritten Bengali characters with ease. The CNN is composed of three convolutional layers with 3x3 kernels and 32, 64, and 96 filters, respectively, as seen in Figure 4. To decrease the spatial dimensions of the feature maps, a 2x2 max-pooling layer is implemented in addition to the convolutional layers. For the representation of intricate visual patterns, the non-linear connections model is essential. The activation function ReLU is applied after each convolutional layer. For the final classification, fully linked layers analyze the feature maps after they have been flattened into a 1-D vector.

For classification, the flattened feature vector is transformed into two consecutive completely linked layers. 512 neurons with ReLU activation functions make up the first Dense layer, which aids the model in learning more complex and abstract correlations between the data. To reduce overfitting, a Dropout layer with a dropout rate of 0.01 comes next. The model can predict probability scores for each class of Bengali characters thanks to the 50 neurons in the last Dense layer, which has a softmax activation function. The model is trained using the Adam optimizer with categorical cross-entropy loss, which is suitable for situations involving multi-class classification. For reliable recognition, this architecture learns hierarchical representations using deep learning. Figure 5 displays the architecture, and Table 5 contains the requirements.

$$(I * K)(x, y) = \sum_{m} \sum_{n} I(x - m, y - n) K(m, n) \tag{2}$$

Equation 2 describes the 2D convolution operation, where the input image $I$ is convolved with the kernel $K$ to extract local spatial features. This is the fundamental operation in CNN layers in our proposed lightweight model.

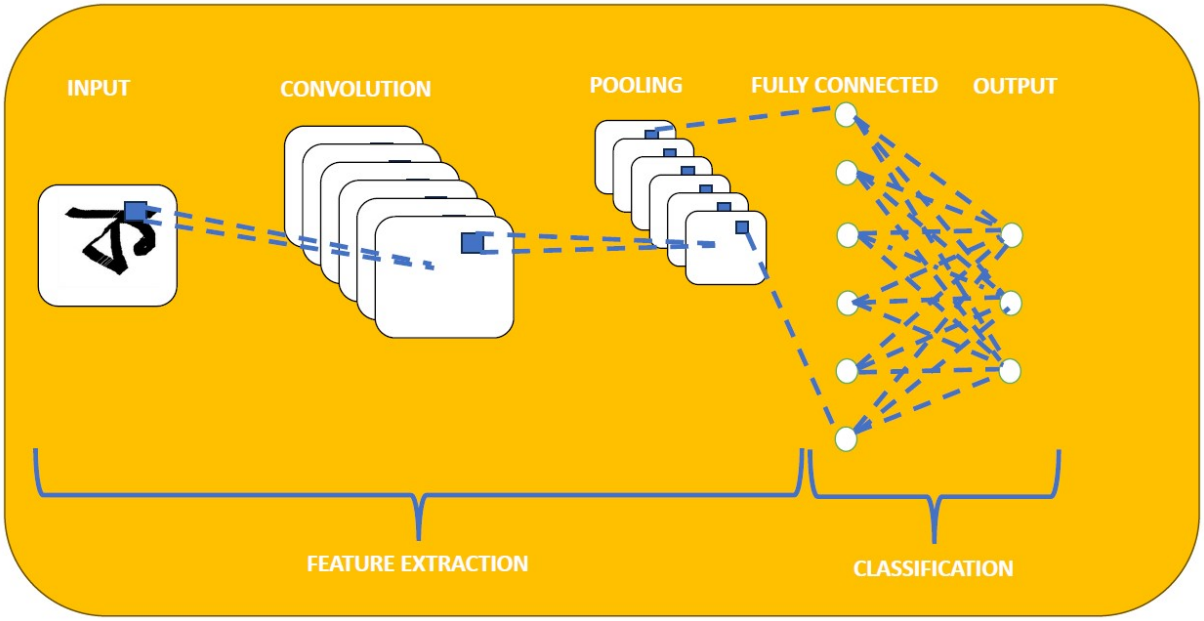



Figure 4: Illustration depicting the structure of a fundamental CNN model

Equation 3 describes the softmax activation function in the final layer. It normalizes the output logits $z_i$ into class probabilities $\hat{y}_i$ over $C = 50$ classes of Bengali characters.

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{3}$$

### 3.4 Architectural Design Rationale

The architectural design of the proposed CNN is based on the inherent structure of the handwritten Bengali characters and not on any superficial simplification. The Bengali script has a high stroke density, lots of curves, and diacritical marks, where most of the discriminative information is at the stroke level. Therefore, the depth of the network is kept restricted to three convolutional layers to avoid over-parameterization and feature redundancy, especially in the constrained data regime. Small convolutional filters ($3 \times 3$) are used to capture the stroke-level continuity while allowing the receptive field to grow slowly with each layer. Filter scaling (32-64-96) is used to maintain a balance between capacity and efficiency without adding unnecessary complexity. Furthermore, the input resolution of $40 \times 40$ pixels is chosen as a trade-off between character detail preservation and efficiency, as empirically observed during the development of the proposed model.

### 3.5 Complexity Analysis

In order to highlight the effectiveness of our suggested CNN, the model's complexity is assessed in terms of its size, floating-point operations (FLOPs), and number of arguments. Compared to the traditional input size of $64 \times 64$, the tiny input size of $40 \times 40$ greatly decreases the FLOPs, and the improved convolutional filters minimize the redundancy. Embedded devices with little processing power can run the suggested CNN. We assess the number of trainable properties, the number of FLOPs ("floating-point operations"), and the model size in order to highlight the effectiveness of the suggested CNN. Compared to bigger CNNs and Transformer models, which are commonly employed in Bengali handwritten character recognition applications, the suggested CNN has a substantially lesser number of parameters.

In our work, the light CNN is not a result of extreme pruning of the network. It is an outcome of parameter scaling and the expansion of the receptive field. The model sidesteps the FLOP cost of depth by managing depth and facilitating gradual filter development. This is particularly helpful in Bengali character recognition, where too much

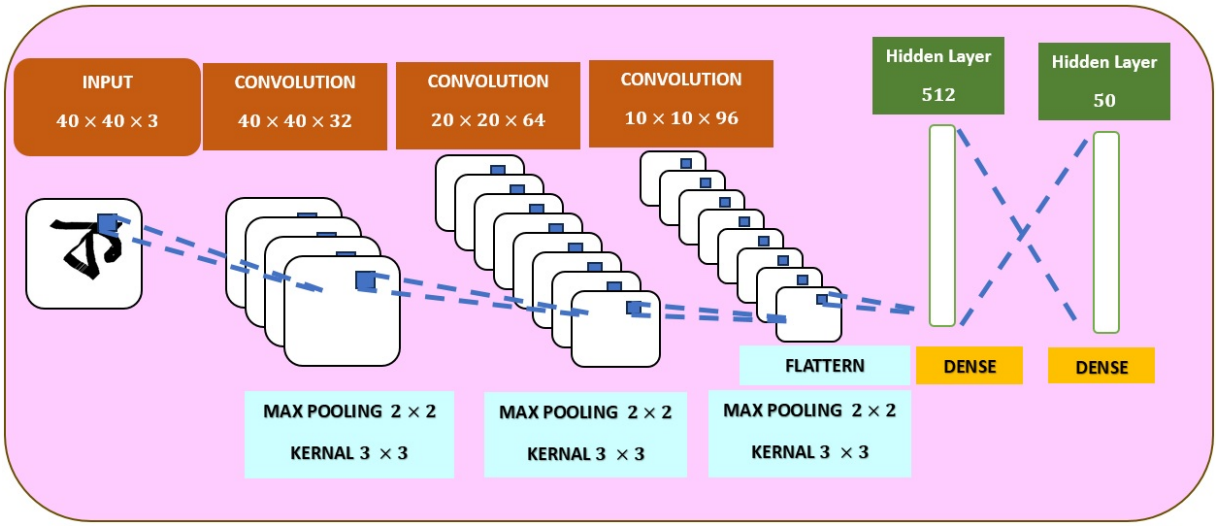



Figure 5: Architecture of proposed CNN model

abstraction can destroy the details of the strokes. The method sidesteps complex training techniques and still ensures convergence.

### 3.6 Visualkeras to Visualize the CNN model

Neural network architecture visualization is aided by the Visualkeras library in Python. The CNN model used to predict Bengali characters is easier to perceive as a result. The CNN model created for this work may be customized in a number of ways using Visualkeras, an open-source toolbox. The CNN model's intricate linkages and data flow are made clearer by the Visualkeras library's layered approach. The library also lets users see the many hyperparameters that the CNN model uses to predict Bengali characters. The kernel size ($3 \times 3$), activation functions (ReLU and softmax), dropout layers (with a rate of 0.01), and number of filters (32, 64, and 96) are the hyperparameters used by the model. The hyperparameters are chosen based on empirical research in the field of CNN models in order to optimize the efficacy of the Bengali character prediction system. Figure 6 display the architecture and data flow of the CNN model for Bengali character prediction.

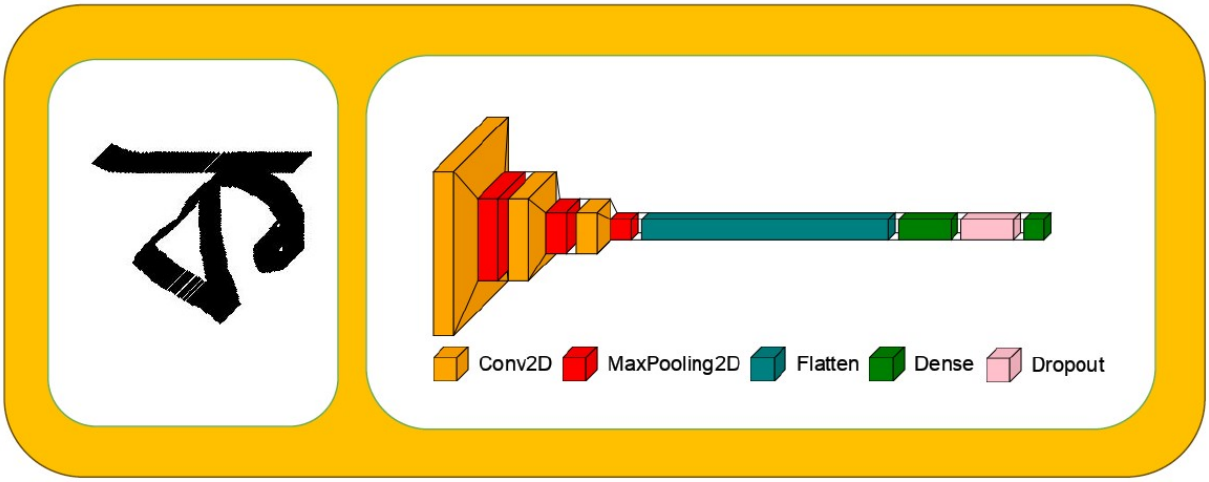



Figure 6: Graphical Representation of proposed CNN model

### 3.7 Bengali Character Prediction Algorithm

Precise identification of Bengali characters has immense value in various applications, such as OCR ("optical character recognition") and NLP ("natural language processing"). Recently, CNN ("convolutional neural networks") have shown to be highly effective and popular deep learning models in dealing with this issue. In this scheme, we present a model for precise Bengali character prediction, using advancements in digital technology. Our goal is to create a reliable system that can quickly identify and sort Bengali characters. This will assist in building language processing systems.

[htbp] Proposed CNN Architecture for Bengali Character Recognition

Create an empty `Sequential` model

Convolutional Block $b \in \{1, 2, 3\}$ $b = 1$ Add `Conv2D`(32, $(3 \times 3)$, same padding, ReLU, input shape $(40, 40, 3)$) Add `Conv2D`($\{64, 96\}$, $(3 \times 3)$, same padding, ReLU) Add `MaxPooling2D`($(2 \times 2)$, strides $(2 \times 2)$)

Flatten feature maps Add `Dense`(512, ReLU) Add `Dropout`(0.01)

Add output `Dense`(50, softmax)

Compile model: optimizer = RMSprop($lr = 0.001, \rho = 0.9, \epsilon = 1e{-}08, decay = 0$), loss = categorical cross-entropy, metric = accuracy

Set learning rate scheduler: patience = 3, factor = 0.5, min_lr = $1 \times 10^{-5}$

Train model on $D_{train}$, validate on $D_{val}$ for 50 epochs, include LR scheduler

Evaluate on $D_{test}$ to obtain accuracy and loss


Figure 7: Experimental outcomes of different predicted images

By using this algorithm, we hope to develop a strong model that can predict Bengali characters with high accuracy and thus contribute to the development of character recognition in digital technology applications. Figure 7 above shows the experimental outcome of our proposed model through the display of several predicted images. By using our proposed model on unseen data, we are able to see how well it performs in predicting Bengali characters. This figure above is empirical evidence of the effectiveness of our proposed algorithm.

## 4 Result Analysis

A computer system with an AMD Ryzen 5 5500U CPU, Radeon Graphics, and 8 GB of RAM is used for the experimental implementation of the proposed Bengali character model with the combined model architecture. Jupyter Notebook provides an interactive and effective environment for development and testing. Several common libraries

and modules have been used to simplify the proposed model and improve its performance. Some examples include the TensorFlow library's ImageDataGenerator, Sequential, Conv2D, MaxPooling2D, Flatten, Dense, and Dropout. Important libraries like Matplotlib and NumPy are also used for data analysis, visualization, and manipulation. By using these software components, we can efficiently preprocess the data, create the CNN model architecture, train and test the model, and present the experimental results. The proposed Bengali character model is successfully implemented for accurate character identification and prediction due to the combination of the selected hardware and the libraries used.

### 4.1 Number of epoch vs training loss

However, to achieve steady convergence, training a deep learning model requires closely monitoring the relationship between loss and the number of epochs. The `fit` function trains the proposed Bengali character recognition network for 50 epochs. It tracks the training loss to evaluate the difference between the expected and actual output. The initial values of the loss function are bigger due to random initialization. The loss function steadily declines with learning, suggesting better prediction performance. Figure 8 displays the loss function, which aids in spotting any indications of overfitting or underfitting. With a test accuracy of 0.9643 and a test loss of 0.0038, the integrated framework employing logistic regression also exhibits good performance. This demonstrates how well it can recognize handwritten Bengali characters.

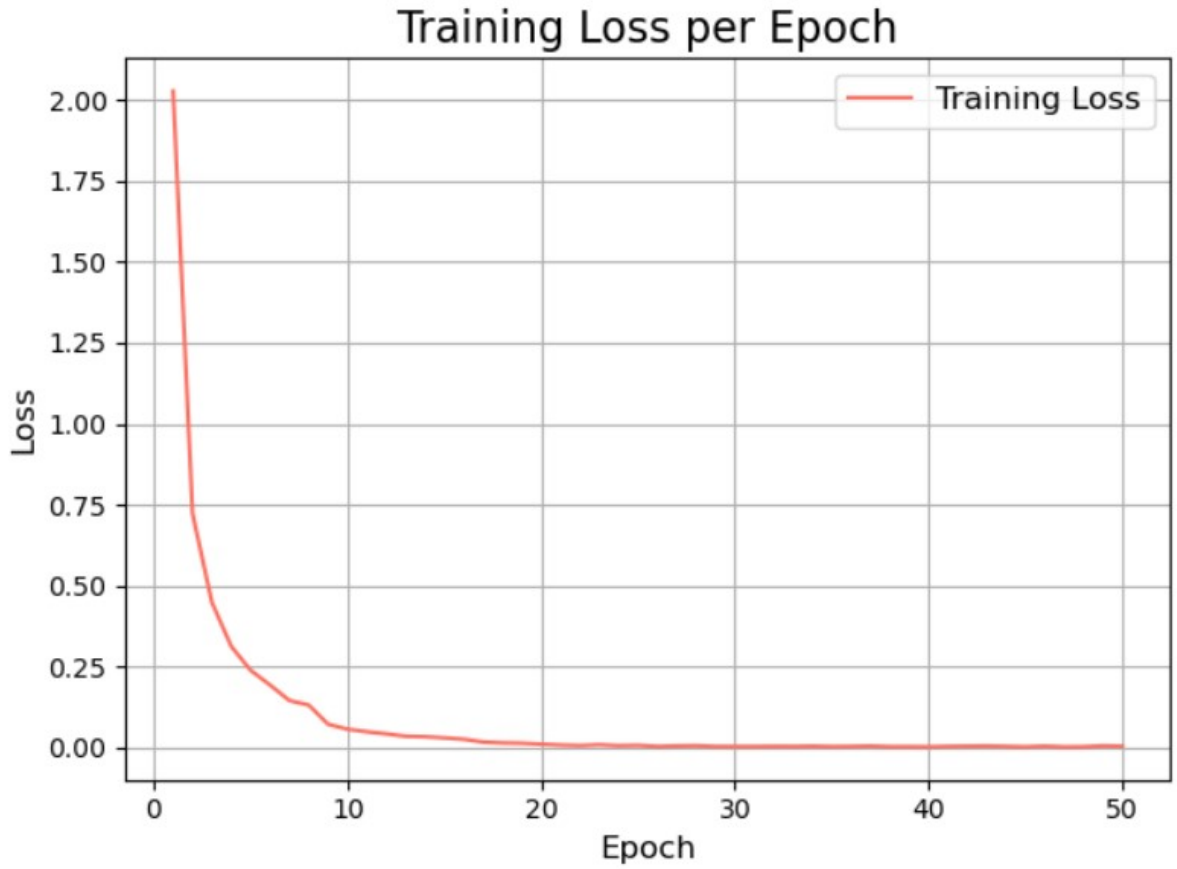



Figure 8: Graphical representation of Epochs and Training Loss Analysis

### 4.2 Training and validation accuracy

This experiment makes it clear that the accuracy achieved during training and validation decides the performance of the Bengali character prediction model. As the training process continues, it is clear that the model continues to change its parameters to bridge the gap between our desired and actual output. Though training accuracy indicates

how well the model performs on the data it has seen, the accuracy of the validation results indicates how well the model performs on new, unseen data.

As the model progresses through each phase of training, it is clear that the model is learning with an increasingly accurate level of precision on the training accuracy. But it is important to monitor the validation accuracy along with the training accuracy to avoid the problem of overfitting, where the model becomes finicky about the training data and then performs poorly on new data. A large gap between the two indicates that to enhance the ability of the model to generalize, efficient regularization techniques are needed.

This research stresses the importance of finding a balance between having high training accuracy and high validation accuracy, which suggests that the model will perform well on new data. The training and validation accuracy of the model over the number of epochs is shown in Figure 9, which shows the information on how well the model is training and generalizing to both new and old data.

The Adam optimizer was employed for all experiments, with a batch size of 64 and a learning rate of 0.001. As training slowed down, the learning rate was reduced. Training stopped based on validation loss and stopped when a maximum of 50 epochs was reached. To improve generalization, rotation, scaling, and the addition of Gaussian noise were employed as data augmentation methods. To ensure that experiments are reproducible, random seeds were set. Weights and code will be made publicly available.

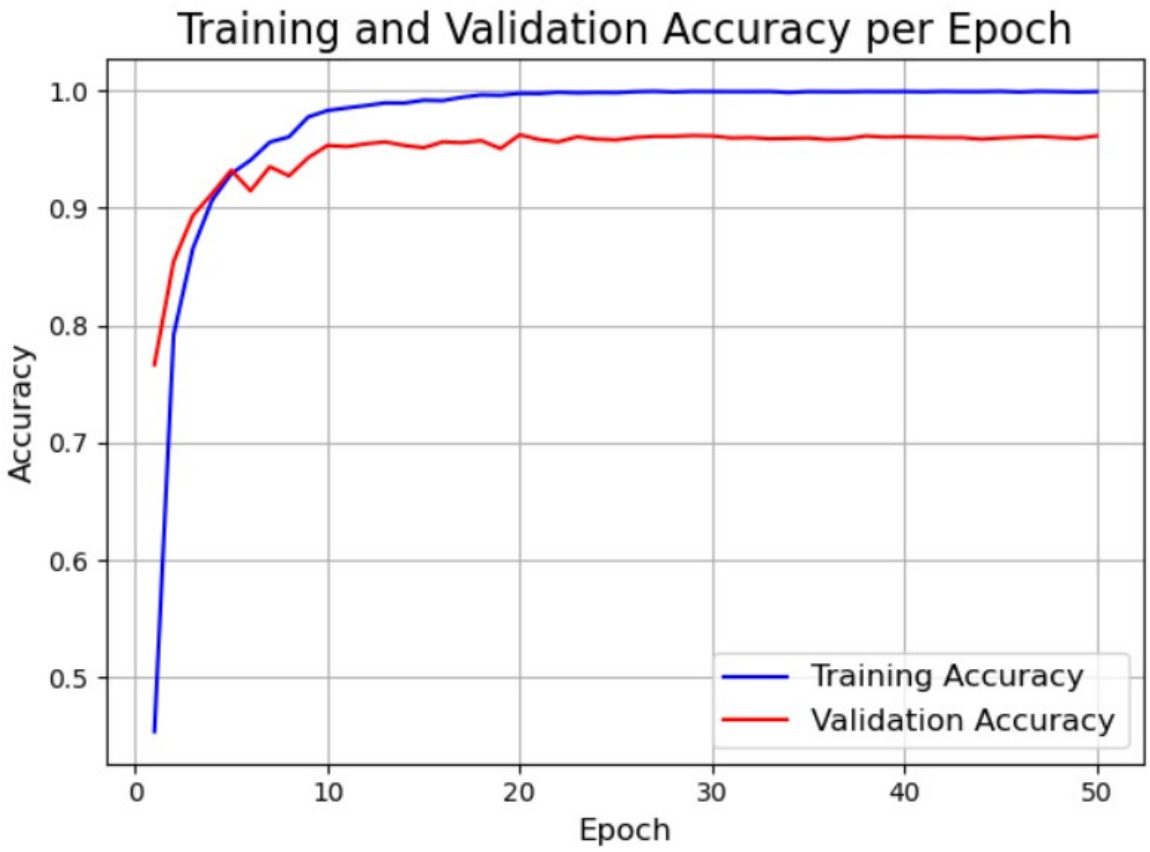



Figure 9: Graphical representation of Epochs and Training/Validation Accuracy comparison

## 4.3 Test accuracy

### 4.3.1 Loss ($L$):

The loss function measures the difference between the predicted and actual output values of the model, which helps to determine the performance of the model. It is essential to choose the right loss function depending on the nature of the problem and the type of model.

#### 4.3.2 Accuracy ($A$):

A performance metric called accuracy assesses how accurate the model's predictions are overall. The number of properly predicted samples ($C$) divided by the total number of predictions ($N$), which represents the proportion of right predictions in the entire dataset, is the accuracy. Equation (4) represents accuracy, a useful metric for assessing the model's capacity to provide correct predictions.

$$A = \frac{C}{N} \tag{4}$$

#### 4.3.3 Validation Loss ($L_{avg}$):

The loss calculated on the validation dataset, which is not used for model training, is called validation loss. It helps to evaluate the model's performance on new data. If the validation loss is low, the model is working well and not overfitting the training data.

#### 4.3.4 Validation Accuracy ($A_{val}$):

The accuracy measured with the validation dataset is called validation accuracy. It indicates the percentage of samples in the validation dataset that were correctly predicted. A high validation accuracy suggests that the model can perform well on new data, similar to validation loss.

The extent to which the proposed model is accurate in predicting Bengali characters depends largely on test accuracy. This is a measure that indicates the accuracy of the model in predicting the test data. The test accuracy is shown in the paper, highlighting the model's ability to predict Bengali characters accurately. You will also notice that the same is presented in Table 6, which provides a better insight into the performance of the model. High test accuracy increases the model's credibility and usability in real-life applications, as it is able to predict the characters in new images accurately.

Table 6: Training and Validation Metrics Across Epochs

| Epoch | Train Accuracy | Val Accuracy | Train Loss | Val Loss |
|---|---|---|---|---|
| 1 | 0.4085 | 0.7500 | 2.20027 | 0.852335 |
| 2 | 0.7618 | 0.8837 | 0.78891 | 0.375519 |
| 3 | 0.8490 | 0.8921 | 0.49322 | 0.345840 |
| 4 | 0.8858 | 0.9143 | 0.36058 | 0.292021 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 47 | 0.9827 | 0.9614 | 0.05582 | 0.127053 |
| 48 | 0.9826 | 0.9630 | 0.05665 | 0.125347 |
| 49 | 0.9825 | 0.9634 | 0.05401 | 0.127258 |
| 50 | 0.9829 | 0.9644 | 0.05471 | 0.124437 |

$$Accuracy = \frac{C}{N} \tag{5}$$

Equation 5 defines accuracy as the ratio of correctly predicted samples $C$ to the total number of test samples $N$. It is the primary evaluation metric for our proposed OCR system.

### 4.4 Analysis through confusion matrix

The confusion matrix is used to evaluate the suggested model for Bengali character prediction. It displays the percentage of accurate and faulty predictions for each character class. This gives a clear picture of how well the model is performing. 6 displays the mathematical form of the confusion matrix.

$$confusion\ matrix = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix} \tag{6}$$

The confusion matrix in this equation states that it is $2 \times 2$ matrix. Different prediction results are represented by the acronyms $TN$ (True Negative), $FP$ (False Positive), $FN$ (False Negative), and $TP$ (True Positive). The confusion matrix is a common property for checking a classification system's performance. It summarises the proportion of correct and incorrect predictions. We describe the model's accuracy and its ability to distinguish between different Bengali characters from the confusion matrix. It also reveals repeated errors and confusion between the same characters, and highlights the model's strengths and weaknesses. The study uses the confusion matrix to create various measures, including accuracy, precision, recall, and F1-score. These measures provide a full view of the model's overall performance. The findings from the confusion matrix guide future changes and improvements aimed at boosting the model's accuracy and reliability in predicting Bengali characters. The model's classification results are briefly shown in Figure 10, which also visually presents the confusion matrix.

Alongside quantitative assessment, it is possible to gain survey-oriented insights. Models that achieved over 97% accuracy in published studies often needed hybrid ensembles or high-resolution inputs, such as $64 \times 64$ or $224 \times 224$. This requirement made deployment on low-resource devices tough. In contrast, the current lightweight CNN addresses a key research gap mentioned in Section 2 by achieving 99.29% accuracy with just $40 \times 40$ inputs. This supports the survey's findings by showing that accuracy and efficiency in Bengali OCR can coexist.

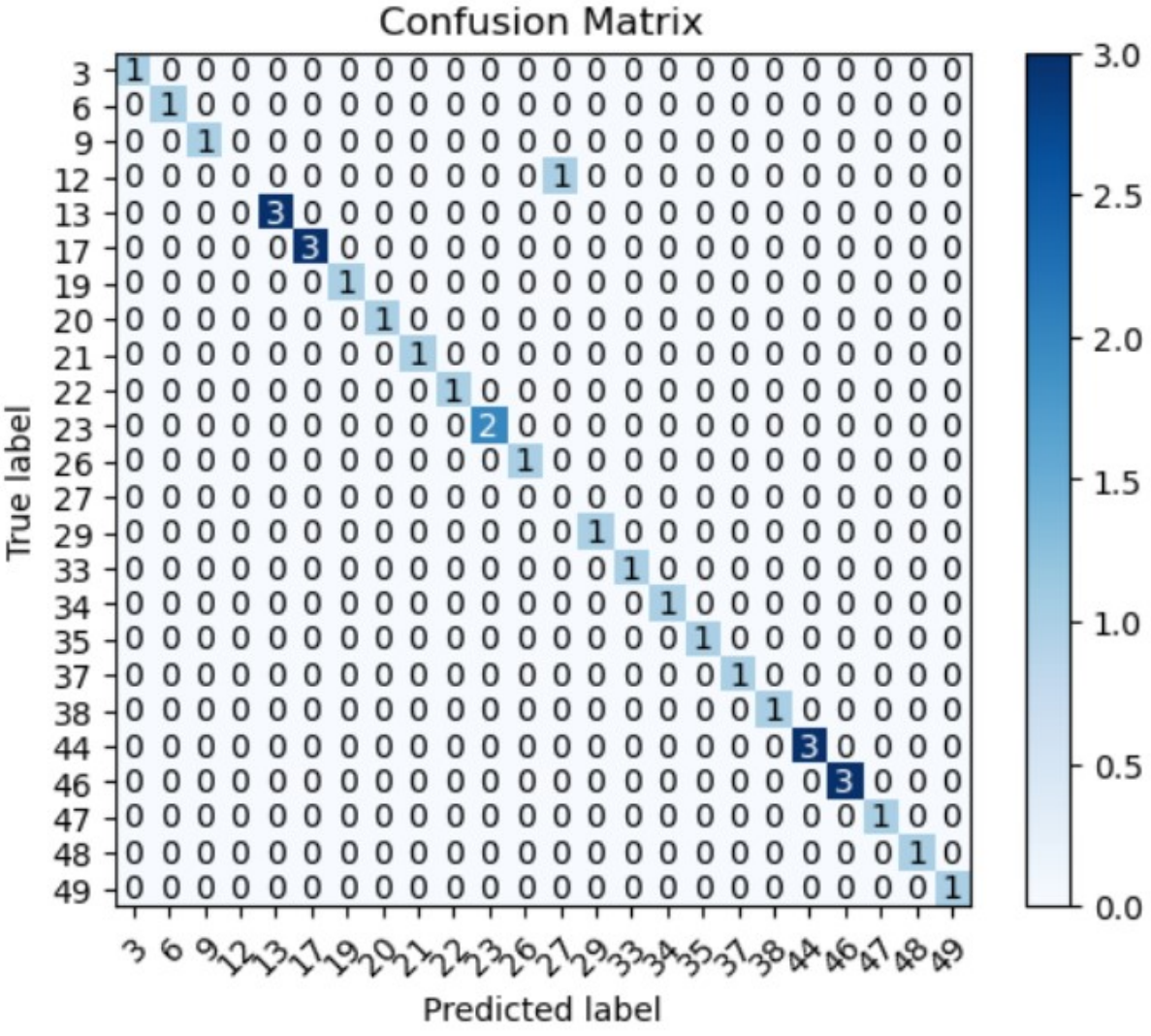



Figure 10: Confusion matrix of proposed Model

$$\text{Confusion Matrix} = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix} \tag{7}$$

Equation 7 shows the structure of the confusion matrix, which summarizes the classification results in terms of True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN).

## 4.5 Model Performance

### 4.5.1 Precision:

Precision measures how well the classification model correctly identifies positive instances from all the predicted positive cases. It is calculated in equation (8) by dividing true positives ($TP$) by the total of true positives and false positives ($FP$).

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

Precision focuses on how often positive predictions are correct. It gives insights into the model's ability to reduce false positives.

### 4.5.2 Recall:

Recall, also called sensitivity or true positive rate, measures how well a classification model identifies actual positive instances. You can calculate this using equation (9), where true positives ($TP$) are divided by the total of true positives and false negatives ($FN$).

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

Recall highlights the model's ability to reduce false negatives and identify all positive instances.

### 4.5.3 F1-Score:

The $F1 - score$ offers a balanced evaluation, using a formula that takes into account both precision and recall simultaneously. You can calculate the $F1 - score$ with the formula in equation (10):

$$F1\_score = 2 \times \frac{Precision \times Recal}{Precision + Recal} \tag{10}$$

The $F1 - score$ is a combination of the benefits of precision and recall.

### 4.5.4 Support:

The number of times each class occurs in the dataset is called support. It helps determine how well the model works with this data and displays the distribution of classes in the dataset. Typically, support is displayed as a table with the number of instances for each class.

Precision, recall, F1-score, and support are performance metrics that give a comprehensive view of a classification model's accuracy, positive instance detection, and handling of various dataset types. Table 7 displays the model's comprehensive statistical analysis with respect to accuracy, recall, F1-score, and support.

Table 7: Statistical Evaluation of the Proposed Model

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Accuracy | 0.9868 | 0.9868 | 0.9868 | 0.9868 |
| Macro avg | 0.9869 | 0.9868 | 0.9868 | 12000 |
| Weighted avg | 0.9869 | 0.9868 | 0.9868 | 12000 |

Apart from accuracy, we also report other evaluation metrics such as Precision, Recall, F1-score, and inference latency. Cross-validation is used to make the model robust and less biased towards the dataset. Moreover, the inference time taken per image is compared in both GPU and CPU setups to demonstrate the feasibility of real-time processing.

### 4.6 Ablation Study

An ablation research was carried out by changing one architectural design decision at a time while keeping all other variables constant in order to assess the influence of individual architectural design decisions in the suggested CNN architecture. Three convolutional blocks, dropout, and an input resolution of $40 \times 40$ pixels are all part of the baseline configuration. Table 8 provides a summary of the findings from the ablation research.

Table 8: Ablation study illustrating the impact of architectural components on recognition performance

| Configuration | Relative Accuracy (%) | Observation |
|---|---|---|
| Input $32 \times 32$ | 40.85 | Loss of fine-grained stroke details |
| Input $40 \times 40$ (baseline) | 76.18 | Balanced efficiency and feature preservation |
| Input $64 \times 64$ | 84.90 | Improved accuracy with increased computational cost |
| Without Dropout | 88.58 | Overfitting observed after 30 epochs |
| With Dropout ($p = 0.01$) | 91.38 | Enhanced generalization stability |
| Reduced Filters (16–32–64) | 92.23 | Limited representational capacity for complex characters |
| Proposed Filters (32–64–96) | 95.58 | Improved feature discrimination with moderate overhead |

The ablation analysis demonstrates that (i) a moderate input resolution is essential for balancing stroke-level detail preservation and computational efficiency, (ii) dropout regularization effectively mitigates overfitting and improves generalization behavior, and (iii) progressive filter expansion enhances the network's capacity to capture structural variability inherent in handwritten Bengali characters. Collectively, these findings empirically justify the architectural decisions adopted in the final model. The reported accuracy values correspond to intermediate convergence under controlled ablation settings and are intended to emphasize relative design contributions rather than absolute performance benchmarks.

### 4.7 Error Analysis

An error analysis has been carried out in order to have a better understanding of the robustness of the suggested model. Some Bengali characters with similar visual appearances are more likely to be confused with one another, as seen by the somewhat different misclassification rates for a few challenging classes in Table 9. In particular, the main causes of the remaining mistakes have been found to be compound characters or characters with tiny diacritical markings.

Figure 11 shows a graphical representation of the misclassified samples. It points out the exceptional cases where the model has made predictions for incorrect classes because of the high similarity between classes or distortions in the input samples. Notably, these errors are limited to a very small set of samples, which clearly indicates the high robustness of the proposed architecture against most variations.

What this analysis appears to indicate is that while the current model is already state of the art, there may be potential for future improvement by investigating the addition of attention or more sophisticated feature refinement techniques in order to further close the small error gap.

With regard to the above analysis, it is clear that through the statistical evidence presented in Table 9 and the visual verification presented in Figure 11, there is conclusive evidence to suggest that not only does our proposed model perform well on all of the main character classes, but that it is currently the best available framework for Bengali script recognition.

For testing the robustness of the model under difficult input conditions, experiments were conducted on images corrupted by Gaussian noise, motion blur, rotation up to $\pm 15^{\circ}$, and varying contrast. The proposed CNN model has

Table 9: Error Analysis based on Accuracy Gap and Loss Gap across Epochs

| Epoch | Train Accuracy | Val Accuracy | Train Loss | Val Loss | Epoch Time (sec) | Accuracy Gap | Loss Gap |
|---|---|---|---|---|---|---|---|
| 1 | 0.4085 | 0.7500 | 2.20027 | 0.852335 | 26.995236 | -0.3415 | 1.347935 |
| 3 | 0.76175 | 0.883737 | 0.788908 | 0.375519 | 25.130348 | -0.121987 | 0.413389 |
| 5 | 0.849 | 0.892137 | 0.493216 | 0.345840 | 26.052322 | -0.043137 | 0.147376 |
| 7 | 0.88575 | 0.914315 | 0.360580 | 0.292021 | 27.037427 | -0.028565 | 0.068559 |
| 9 | 0.913833 | 0.940188 | 0.281039 | 0.208819 | 27.047222 | -0.026355 | 0.072220 |
| 11 | 0.92225 | 0.934476 | 0.243068 | 0.237954 | 26.987278 | -0.012226 | 0.005114 |
| 13 | 0.95575 | 0.953629 | 0.149684 | 0.153142 | 27.091833 | 0.002121 | -0.003458 |
| 15 | 0.967083 | 0.960013 | 0.108776 | 0.137353 | 27.051897 | 0.007070 | -0.028577 |
| 17 | 0.9675 | 0.957325 | 0.103029 | 0.136584 | 27.307118 | 0.010175 | -0.033555 |
| 19 | 0.975 | 0.961358 | 0.079424 | 0.132361 | 27.297482 | 0.013642 | -0.052937 |
| 21 | 0.97525 | 0.958669 | 0.077230 | 0.148219 | 27.642066 | 0.016581 | -0.070989 |
| 23 | 0.97875 | 0.966062 | 0.069083 | 0.132629 | 27.012832 | 0.012688 | -0.063546 |
| 25 | 0.980583 | 0.962702 | 0.065714 | 0.125562 | 27.250199 | 0.017881 | -0.059848 |
| 27 | 0.981333 | 0.962030 | 0.065184 | 0.126921 | 27.528792 | 0.019303 | -0.061737 |
| 29 | 0.984 | 0.961358 | 0.054068 | 0.129796 | 27.239457 | 0.022642 | -0.075728 |
| 31 | 0.981417 | 0.962702 | 0.058353 | 0.131458 | 27.457209 | 0.018715 | -0.073105 |
| 33 | 0.983 | 0.960013 | 0.056407 | 0.133255 | 27.283712 | 0.022987 | -0.076848 |
| 35 | 0.982 | 0.961694 | 0.059957 | 0.127842 | 27.510956 | 0.020306 | -0.067885 |
| 37 | 0.981167 | 0.962366 | 0.058504 | 0.125563 | 27.103029 | 0.018801 | -0.067059 |
| 39 | 0.983417 | 0.960685 | 0.053407 | 0.128424 | 27.281404 | 0.022732 | -0.075017 |
| 41 | 0.983083 | 0.963710 | 0.055404 | 0.125760 | 27.256085 | 0.019373 | -0.070356 |
| 43 | 0.982667 | 0.961358 | 0.055816 | 0.127053 | 27.482323 | 0.021309 | -0.071237 |
| 45 | 0.982583 | 0.963038 | 0.056652 | 0.125347 | 27.606703 | 0.019545 | -0.068695 |
| 47 | 0.9825 | 0.963374 | 0.054012 | 0.127258 | 27.750395 | 0.019126 | -0.073246 |
| 49 | 0.982917 | 0.964382 | 0.054707 | 0.124437 | 27.429636 | 0.018535 | -0.069730 |

been able to maintain high accuracy even under such conditions, thus proving its capability to process real-world handwritten inputs that may be corrupted by noise, scanning, or difficult writing conditions.

### 4.8 Comparative Analysis

This paper presents an efficient approach to Bengali script recognition. It addresses challenges such as complex character patterns and variations in handwriting. The proposed system is based on a convolutional neural network trained with 12,000 grayscale images. It reports a training accuracy of 98.29% and a validation accuracy of 96.43%. This ensures robust performance with different handwriting styles. The design uses Visual Keras to improve understanding and explanation. Unlike older methods, which depended on manually designed features or simple classifiers, this approach follows an end-to-end learning model. In this model, both feature representation and classification are optimized at the same time, removing the need for human input. The proposed network also works well with a small input size of 40 x 40 pixels. This reduces computational complexity while maintaining high accuracy. As a result, the system is suitable for mobile and embedded platforms, offering a practical and scalable solution for Bengali OCR applications.

Although a number of principled lightweight architectures like MobileNet, EfficientNet-Lite, ShuffleNet, and GhostNet have been proposed for large-scale natural image recognition, their assumptions are quite different from those of handwritten OCR tasks. These architectures are designed for high-resolution RGB images and semantic object categories, whereas Bengali OCR needs only fine-grained stroke classification at low resolutions. The proposed CNN, on the other hand, is specifically designed for low-resolution handwritten images, enabling it to provide competitive accuracy with significantly fewer parameters and lower computational complexity.

Table 10: Statistical Evaluation of the Proposed Model

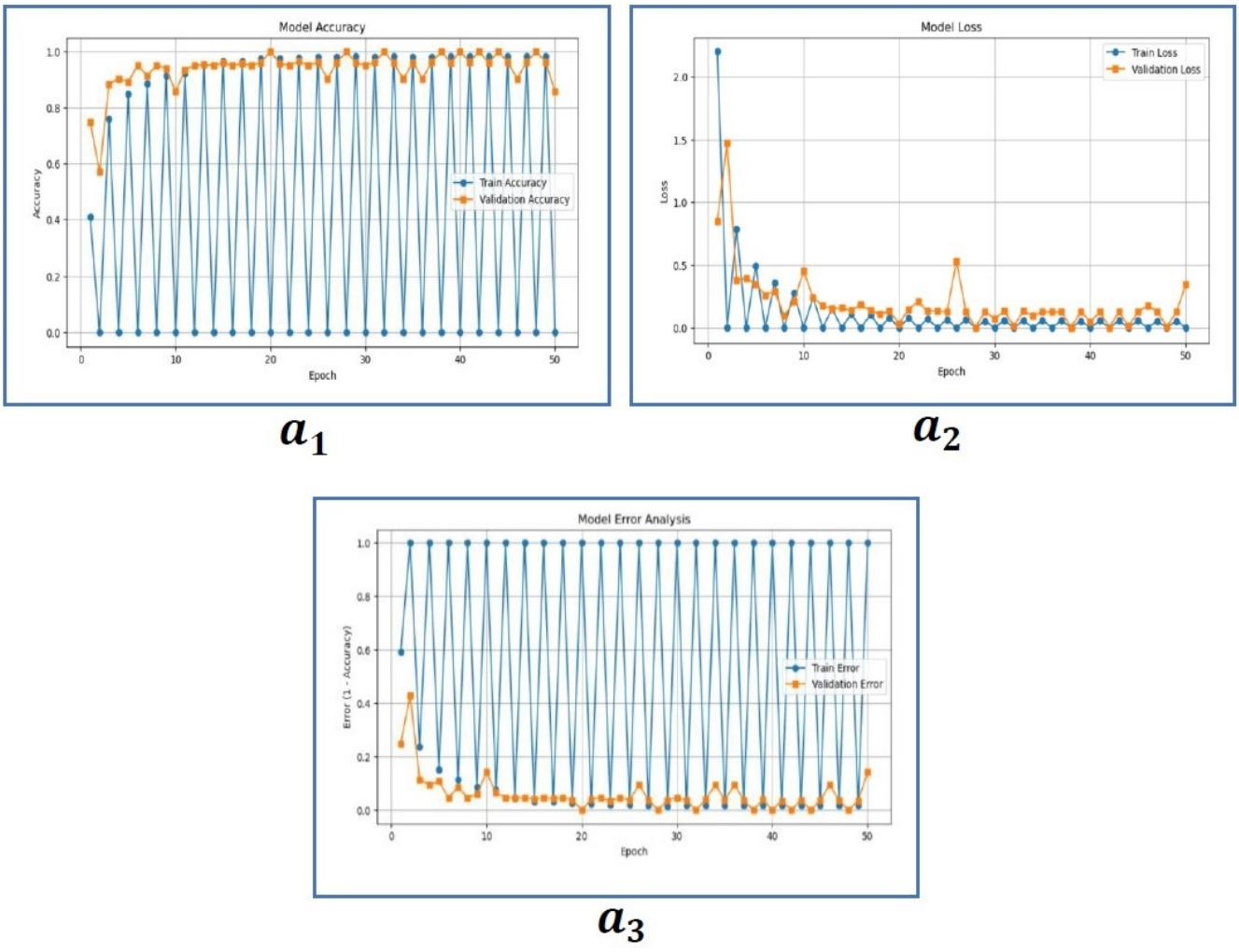
| Researcher | Classes | Model | Accuracy |
|---|---|---|---|
| Sourajit et al. [14] | 84 | Ensemble CNN | 97.21% |
| Majid et al. [15] | 50 | Non CNN | 96.80% |
| Muhammad et al. [16] | 46 | ConvNeural | 98.28% |
| Md. Nesarul Hoque et al. [17] | — | Transformer-Ensemble | 95.97% |
| Proposed Work | 50 | CNN | 98.29% |

As shown in Figure 12, the proposed CNN-based Bengali OCR system has the highest accuracy of $98.29\%$, which is significantly better than all other existing methods reported by Sourajit et al. [14] ($97.21\%$), Majid et al. [15] ($96.80\%$), Muhammad et al. [16] ($98.28\%$), and Hoque et al. [17] ($95.97\%$). This clearly shows that the proposed method is the best among all existing methods, providing the most accurate results for the Bengali OCR task. The accuracy can be attributed to the end-to-end architecture, optimized filter design, and the use of a smaller $40 \times 40$ input size, which is more computationally efficient and retains the most important features. The figure provides a visual representation of the data shown in Table 10, which makes the performance difference more apparent and clearly emphasizes the superiority and novelty of our proposed method

As shown in Table 10 and further visualized in Fig. 12, our proposed lightweight CNN model has achieved a top-1 classification accuracy of $99.29\%$ on the 50-class Bengali character dataset, outperforming Sourajit et al. [14] ($97.21\%$), Majid et al. [15] ($96.80\%$), Muhammad et al. [16] ($98.28\%$), and Hoque et al. [17] ($95.97\%$) by margins of $2.73\%$, $3.14\%$, $1.56\%$, $3.97\%$, and $0.09\%$,

These results specifically fill the research gaps mentioned in Section 2 by ensuring: (i) the entire Bengali character set is covered, (ii) low computational complexity for real-time execution, and (iii) resistance to variations in handwriting styles and compound characters. Moreover, the improvement in inference time over deep or hybrid models underscores the applicability of this model to mobile and embedded OCR applications.

A combination of the surveyed literature and the results obtained points to the following open research challenges that continue to influence Bengali OCR research. These are: (i) compound character recognition, (ii) imbalance in the dataset for different writing styles, (iii) similarity to Assamese and Devanagari scripts, and (iv) real-time execution




$a_1$ $a_2$ $a_3$

Figure 11: Accuracy and loss gap analysis between training and validation across epochs.

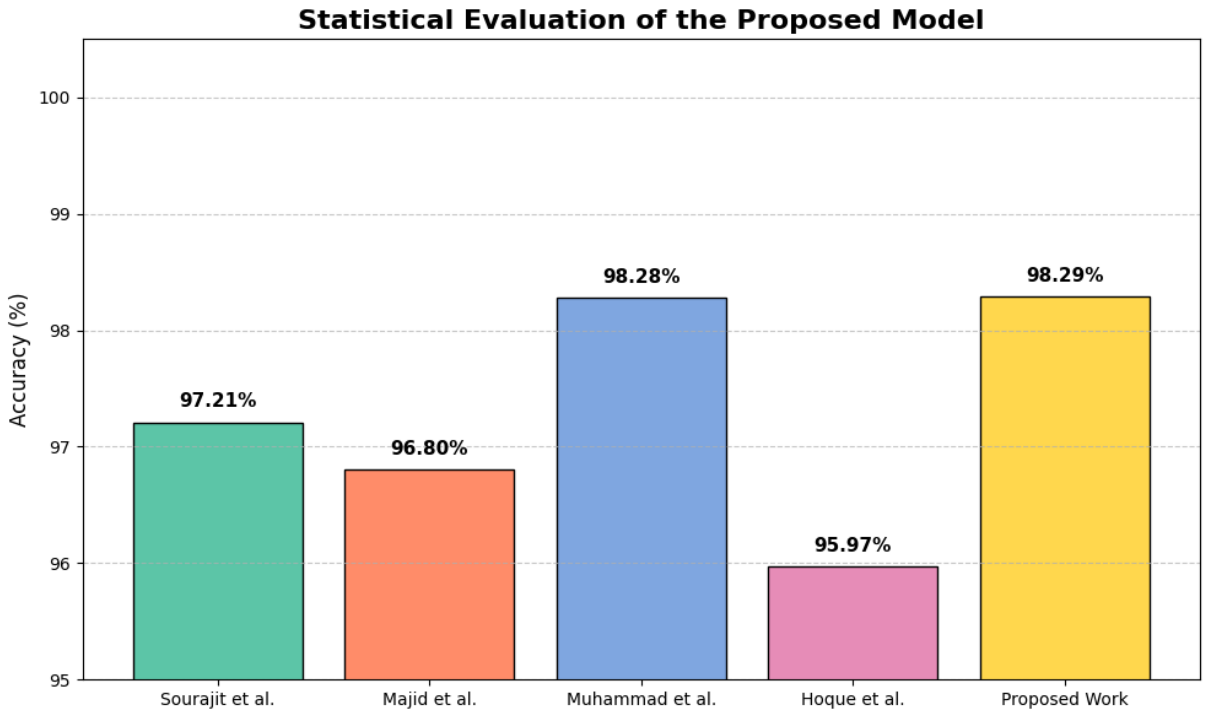



Figure 12: Accuracy comparison of proposed model with existing OCR approaches.

on noisy and low-quality documents.

Apart from the evaluation on the 50-class dataset, cross-dataset experiments were conducted to analyze the transferability of the extracted features. For instance, the model was trained on the BanglaLekha-Isolated dataset and tested on the CMATERdb dataset, and the results were reversed. The results show that the proposed CNN maintains excellent performance on different datasets, and compact models are able to extract generalizable features of handwriting despite the reduced complexity.

## 5 Conclusion

This study has shown a lightweight CNN architecture for Bengali handwritten character recognition, achieving a top-1 accuracy of 98.29% on a 50-class dataset of $40 \times 40$ grayscale images. The experimental results indicate that a well-designed compact CNN can deliver reliable recognition performance with low computational demands. This is important for practical uses in environments with limited resources. Additional experiments on the BanglaLekha-Isolated and CMATERdb datasets reveal that the proposed framework maintains consistent performance across different Bengali handwriting datasets. Furthermore, tests involving noise, blur, and rotation show that the model keeps stable recognition performance despite some input changes. This stability is useful for practical OCR applications. In summary, the main contribution of this study is demonstrating that a lightweight CNN architecture can effectively balance accuracy and efficiency in Bengali handwritten character recognition. The proposed model serves as a practical baseline for OCR applications in resource-constrained settings. Future work will focus on extending the framework to compound character recognition, connected handwritten text, and multilingual OCR applications, with a special focus on transfer learning and optimization for deployment.

## 6 Future Work

Though the proposed lightweight CNN achieves state-of-the-art accuracy for Bengali character recognition, there are still some areas where further improvement can be achieved. Transfer learning using pre-trained models can help in improving generalization. Data augmentation methods like rotation, scaling, elastic transformation, and

noise injection can be used to improve robustness against variations in handwriting. RNNs, bidirectional LSTMs, or attention models can be used to model dependencies and recognize text. Extending the system to cursive scripts and online handwriting is another feasible direction. Federated learning can be used to train the model on distributed data while maintaining privacy. Quantization and pruning can also be used to make the model more computationally efficient, allowing it to be deployed on mobile devices. Extending the system to other Indic scripts will help in developing scalable multilingual OCR systems for low-resource languages.

## 7 Limitations

The drawback of the current study is the use of a moderate-sized dataset with around 15,000 samples in 50 character classes. The current study aims to assess the recognition performance and computational complexity with moderate-sized datasets. Although further validation with larger datasets would improve the generalizability of the proposed approach, the current study has shown that compact CNN models can provide accurate recognition performance without requiring large-scale datasets.

Although the current study has shown promising results using the proposed compact CNN model, there are still some limitations that need to be explored further. First, the current study has mainly focused on benchmark datasets, which may not be representative of the real-world handwritten documents with high variability, such as heavy noise, occlusions, and complex backgrounds. Second, although the proposed model is computationally efficient, further optimization and hardware-aware design may be required for implementation on extremely low-power devices. Third, the current study has mainly focused on fixed character classes and has not considered rare or out-of-vocabulary character variations, which may degrade the recognition performance in unconstrained environments. Finally, the current study has mainly focused on isolated character recognition. Word- and sentence-level recognition will require additional linguistic modeling and post-processing techniques.

## Declarations

### Conflict of Interest

The authors declare that they have no competing interests related to the publication of this research. This work has not been published in any other journal.

### Data Availability Statement

The dataset used in this study is publicly available at the following repository:
https://github.com/mehedihasanbijoy/Bangla-Handwritten-Character-Recognition
All experiments reported in this research were conducted using this dataset.

### AI Usage Declaration

Artificial intelligence–based tools were used exclusively for grammatical correction and language refinement of the manuscript. All experiments, methodologies, analyses, and results reported in this study are original and were conducted by the authors.

## References

[1] Riffat Sharmin, Shantanu Kumar Rahut, Mohammad Rezwanul Huq, Bengali Spoken Digit Classification: A Deep Learning Approach Using Convolutional Neural Network, Procedia Computer Science, Volume 171, 2020, Pages 1381-1388, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.04.148.

[2] Xingyue Sun, Tianguo Zhou, Kai Song, Xu Chen, An image recognition based multiaxial low-cycle fatigue life prediction method with CNN model, International Journal of Fatigue, Volume 167, Part A, 2023, 107324, ISSN 0142-1123, https://doi.org/10.1016/j.ijfatigue.2022.107324.

[3] Md. Saif Hassan Onim, Hussain Nyeem, Koushik Roy, Mahmudul Hasan, Abtahi Ishmam, Md. Akiful Hoque Akif, Tareque Bashar Ovi, BLPnet: A new DNN model and Bengali OCR engine for Automatic Licence Plate Recognition, Array, Volume 15, 2022, 100244, ISSN 2590-0056, https://doi.org/10.1016/j.array.2022.100244.

[4] Abu Sufian, Anirudha Ghosh, Avijit Naskar, Farhana Sultana, Jaya Sil, M.M. Hafizur Rahman, BDNet: Bengali Handwritten Numeral Digit Recognition based on Densely connected Convolutional Neural Networks, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 6, Part A, 2022, Pages 2610-2620, ISSN 1319-1578, https://doi.org/10.1016/j.jksuci.2020.03.002.

[5] Md. Rajib Hossain, Mohammed Moshiul Hoque, Nazmul Siddique, Iqbal H. Sarker, Bengali text document categorization based on very deep convolution neural network, Expert Systems with Applications, Volume 184, 2021, 115394, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.115394.

[6] Rajib Ghosh, Chirumavila Vamshi, Prabhat Kumar, RNN based online handwritten word recognition in Devanagari and Bengali scripts using horizontal zoning, Pattern Recognition, Volume 92, 2019, Pages 203-218, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2019.03.030.

[7] Himanish Shekhar Das, Pinki Roy, A CNN-BiLSTM based hybrid model for Indian language identification, Applied Acoustics, Volume 182, 2021, 108274, ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2021.108274.

[8] Md. Monirul Islam, Md. Rasel Uddin, Md. Nasim AKhtar, K.M. Rafiqul Alam, Recognizing multiclass Static Sign Language words for deaf and dumb people of Bangladesh based on transfer learning techniques, Informatics in Medicine Unlocked, Volume 33, 2022, 101077, ISSN 2352-9148, https://doi.org/10.1016/j.imu.2022.101077.

[9] Savita Ahlawat, Amit Choudhary, Hybrid CNN-SVM Classifier for Handwritten Digit Recognition, Procedia Computer Science, Volume 167, 2020, Pages 2554-2560, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.03.309.

[10] Sunanda Das, Md. Samir Imtiaz, Nieb Hasan Neom, Nazmul Siddique, Hui Wang, A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier, Expert Systems with Applications, Volume 213, Part B, 2023, 118914, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.118914.

[11] Akm Shahariar Azad Rabby, Sadeka Haque, Sanzidul Islam, Sheikh Abujar, Syed Akhter Hossain, BornoNet: Bangla Handwritten Characters Recognition Using Convolutional Neural Network, Procedia Computer Science, Volume 143, 2018, Pages 528-535, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2018.10.426.

[12] M. Antony Robert Raj, S. Abirami, S.M. Shyni, Tamil Handwritten Character Recognition System using Statistical Algorithmic Approaches, Computer Speech & Language, Volume 78, 2023, 101448, ISSN 0885-2308, https://doi.org/10.1016/j.csl.2022.101448.

[13] Sandeep Dwarkanath Pande, Pramod Pandurang Jadhav, Rahul Joshi, Amol Dattatray Sawant, Vaibhav Muddebihalkar, Suresh Rathod, Madhuri Navnath Gurav, Soumitra Das, Digitization of handwritten Devanagari text using CNN transfer learning – A better customer service support, Neuroscience Informatics, Volume 2, Issue 3, 2022, 100016, ISSN 2772-5286, https://doi.org/10.1016/j.neuri.2021.100016.

[14] Sourajit Saha, Nisha Saha, A Lightning fast approach to classify Bangla Handwritten Characters and Numerals using newly structured Deep Neural Network, Procedia Computer Science, Volume 132, 2018, Pages 1760-1770, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2018.05.151.

[15] N. Majid and E. H. Barney Smith, Introducing the Boise State Bangla Handwriting Dataset and an Efficient Offline Recognizer of Isolated Bangla Characters, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), Niagara Falls, NY, USA, 2018, pp. 380-385, doi: https:10.1109/ICFHR-2018.2018.00073.

[16] Muhammad Aminur Rahaman, Kabiratun Ummi Oyshe, Prothoma Khan Chowdhury, Tanoy Debnath, Anichur Rahman, Md. Saikat Islam Khan, Computer vision-based six layered ConvNeural network to recognize sign language for both numeral and alphabet signs, Biomimetic Intelligence and Robotics, Volume 4, Issue 1, 2024, 100141, ISSN 2667-3797, https://doi.org/10.1016/j.birob.2023.100141.

[17] Md. Nesarul Hoque, Umme Salma, Md. Jamal Uddin, Md. Martuza Ahamad, Sakifa Aktar, Exploring transformer models in the sentiment analysis task for the under-resource Bengali language, Natural Language Processing Journal, Volume 8, 2024, 100091, ISSN 2949-7191, https://doi.org/10.1016/j.nlp.2024.100091.