I. J. Engineering and Manufacturing, 2026, 2, 111-131
Published Online on April 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijem.2026.02.07

Deep Learning-Based Recognition of Handwritten Devanagari Multilevel Compound Characters: A Performance Comparison

Ashwini B. Patil*
Department of Computer Science & Engineering, RKDF University, Bhopal (MP), India
E-mail: ash82patil@gmail.com
ORCID ID: https://orcid.org/0000-0003-3387-5543
*Corresponding Author

Puneet Dwivedi
Department of Computer Science and Engineering, RKDF University, Bhopal (MP), India
E-mail: puneet86.dwivedi@gmail.com
ORCID ID: https://orcid.org/0009-0008-4185-3575

Received: 27 June, 2025; Revised: 14 October, 2025; Accepted: 16 January, 2026; Published: 08 April, 2026

Abstract: Handwritten character recognition is a crucial challenge in artificial intelligence and computer vision, particularly for complex scripts like Devanagari. Devanagari is widely used in Hindi, Marathi, and Nepali and consists of intricate multilevel compound characters, ligatures, and highly variable handwriting styles. Despite advances in optical character recognition (OCR) technology, accurately recognizing handwritten Devanagari characters remains difficult. This study compares various deep learning models, including Convolutional Neural Networks (CNN), CNN-SVM, Long Short-Term Memory (LSTM), EfficientNet, and a newly proposed attention-based CNN model. Extensive experiments were conducted on a diverse dataset containing simple and compound handwritten Devanagari characters. The proposed attention-based CNN model outperforms traditional methods, achieving a recognition accuracy of 96.5%, significantly higher than CNN (88.0%), CNN-SVM (88.5%), LSTM (92.0%), and EfficientNet (93.0%). The study employs advanced data augmentation techniques to enhance model robustness, making it adaptable to various handwriting styles. The attention mechanism in the proposed CNN model allows for improved feature extraction, leading to higher recognition accuracy and efficiency. This research contributes to developing robust OCR systems for the Devanagari script, enabling improved digitisation and preservation of Indian languages. The proposed approach can be extended to other complex scripts like Bengali and Tamil, further advancing multilingual OCR technologies, is now exploratory and has not been experimentally verified in this study. Future work can address the need for thorough cross-script evaluation and transfer learning studies to verify the adaptability of the attention-based CNN architecture, despite its inherent script-agnostic nature. The findings of this study hold significant implications for text digitization, historical document preservation, and automated language processing applications.

Index Terms: Handwritten Character Recognition, Devanagari Script, Optical Character Recognition (OCR), CNN, Attention-based CNN, LSTM, Compound Characters, Deep Learning, Machine Learning, Data Augmentation.

1. Introduction

Handwritten character recognition is a critical area in artificial intelligence and computer vision, with widespread applications in document digitization, automated text processing, and archival preservation [1]. While Optical Character Recognition (OCR) has advanced significantly in recognizing printed text, handwritten text recognition, especially for complex scripts like Devanagari, remains a significant challenge [2,3]. The Devanagari script is widely used in languages such as Hindi, Marathi, and Nepali. Unlike Latin scripts, Devanagari consists of multilevel compound characters, ligatures, and diacritics, which make recognition particularly difficult [4]. Devanagari recognition is essential for various applications, including historical document preservation, automated form processing, and banking. However, the complexity of the script poses significant difficulties for OCR systems. Traditional OCR technologies struggle with accurately segmenting and classifying Devanagari characters due to the presence of overlapping strokes,

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

contextual modifications, and variability in handwriting styles. Unlike printed text, handwritten characters exhibit irregularities such as different stroke thicknesses, variable spacing, and unique personal writing styles, making recognition even more challenging.

Character recognition is the process of converting printed or handwritten text into a machine-readable format. This technology has applications in various fields, such as document scanning, digital libraries, automated postal systems, and banking [5]. OCR systems are primarily designed for printed text recognition, but handwritten text recognition remains challenging due to the diversity in handwriting styles and character formations. A typical OCR system consists of four main components: image preprocessing, feature extraction, pattern classification, and text reconstruction. Image preprocessing involves binarization, noise reduction, skew correction, and character segmentation to enhance the quality of input images. Feature extraction identifies distinguishing characteristics such as edges, curves, and strokes, enabling the system to differentiate between characters. The classification step employs machine learning or deep learning models to recognize characters by comparing extracted features against stored templates. Finally, post-processing techniques like natural language processing (NLP) improve the accuracy of the recognized text by correcting errors and refining sentence structure [6]. Despite advancements in OCR technology, recognizing handwritten text, particularly in Devanagari, remains a major challenge due to script complexity, handwriting variability, and segmentation difficulties. Addressing these challenges requires innovative approaches that incorporate deep learning techniques for better accuracy and efficiency.

Devanagari script introduces multiple challenges for OCR systems, making handwritten text recognition significantly more complex than Latin-based scripts. One of the primary challenges is the presence of multilevel compound characters, where multiple consonants combine to form a single character [7]. These "conjuncts" create unique visual patterns that differ from standalone characters, making segmentation and classification difficult. Another major challenge is using ligatures and diacritics (matras) to modify consonants. Unlike English, where vowels exist as separate characters, Devanagari vowels attach to consonants in different positions, changing their shape. Recognizing these variations accurately requires an OCR system capable of contextual interpretation. Handwriting variability further complicates recognition. Unlike printed text, handwritten characters exhibit inconsistent stroke thickness, slant, spacing, and alignment. This variability makes developing a universal model that performs well across different writing styles challenging. Additionally, segmentation difficulties arise due to Devanagari's characteristic Shirorekha (horizontal line connecting characters), making distinguishing individual characters in a word difficult [8].

The availability of limited annotated datasets is another barrier to practical model training. Deep learning models require large amounts of labeled data to achieve high accuracy, but publicly available handwritten Devanagari datasets are scarce compared to Latin-based scripts. Lastly, OCR systems must be robust to noise and distortions caused by ink smudges, poor handwriting, or low-resolution images, all degrading recognition performance [9].

Traditional OCR systems relied on classical machine learning techniques such as Support Vector Machines (SVM) and k-nearest Neighbors (k-NN) for character classification. These methods required handcrafted feature extraction, which limited their adaptability to different writing styles and complex scripts. While these techniques were moderately successful for printed text, they struggled with handwritten Devanagari characters [10]. With advancements in deep learning, character recognition has significantly improved. Convolutional Neural Networks (CNNs) have proven effective for handwritten character recognition by automatically learning hierarchical features from images. However, traditional CNNs sometimes fail to capture long-range dependencies in characters, especially in scripts with complex ligatures.

Researchers have developed hybrid models such as CNN-SVM to overcome these limitations, where CNN extracts features while SVM performs classification. Additionally, Long Short-Term Memory (LSTM) networks have been explored for recognizing sequential patterns in handwritten text. While LSTM models improve contextual understanding, they are computationally intensive. Another promising approach is EfficientNet, a deep learning model that optimizes performance using compound scaling. However, even advanced CNN and LSTM models struggle with distinguishing intricate details in handwritten Devanagari characters. Attention-based CNN models introduce a novel approach by enabling the network to focus on significant regions of the character, enhancing feature extraction and improving recognition accuracy.

The increasing demand for OCR solutions in languages like Hindi, Marathi, and Nepali underscores the need for accurate Devanagari character recognition. As digitization efforts expand, the ability to process handwritten documents efficiently is crucial for government institutions, educational organizations, and historical archives [11]. One of the primary motivations for this research is the digitization of historical documents, where ancient manuscripts and handwritten records must be converted into searchable text. Automated document processing in banking, government services, and legal documentation can streamline workflows and reduce manual data entry errors.

Another critical application is assistive technology for visually impaired individuals, where OCR systems can enable text-to-speech conversion, making Devanagari content more accessible. Moreover, enhancing searchability in digital archives by indexing handwritten text will significantly benefit researchers and scholars [12]. Given the complexity of the Devanagari script, there is a strong need to develop a robust OCR system capable of handling handwriting variability, compound characters, and contextual modifications. This study aims to bridge this gap by evaluating different deep-learning models and introducing an attention-based CNN architecture for improved accuracy.

This study makes several key contributions to the field of handwritten character recognition:

- Comparative Analysis: A detailed evaluation of CNN, CNN-SVM, LSTM, EfficientNet, and attention-based CNN models for Devanagari OCR.
- Proposed Attention-based CNN Model: Development of a novel architecture that enhances feature extraction and improves recognition accuracy.
- Dataset Augmentation: Application of advanced augmentation techniques to improve model generalization.
- Optimization for OCR Applications: Improve character segmentation, classification, and overall OCR accuracy for handwritten Devanagari script.

Handwritten Devanagari character recognition presents unique challenges due to script complexity, handwriting variability, and segmentation difficulties. While traditional machine learning methods have limitations, deep learning techniques offer promising solutions. This study aims to evaluate different models and introduce an improved attention-based CNN architecture, contributing to the advancement of OCR technology for the Devanagari script. The findings of this research will have significant implications for document digitization, historical text preservation, and multilingual OCR applications.

This study focuses specifically on isolated handwritten multilevel compound characters, which represent the most challenging recognition unit of multilevel compound characters in Devanagari script. While word and line OCR system are essential for end-end applications. The accurate isolated character recognition is foundation part of higher level recognition system.

2. Literature Review

Handwritten character recognition has been an area of significant research interest in artificial intelligence (AI) and computer vision. The recognition of the Devanagari script poses unique challenges due to the presence of compound characters, ligatures, and diacritics. This section presents an overview of previous research in handwritten character recognition, with a focus on machine learning, deep learning, and hybrid approaches.

Sheikh Mohammad Jubaer et al. [13] introduce BN-DRISHTI, which combines YOLO with Hough and Affine transformation for Bangla handwritten text segmentation. BN-DRISHTI achieved high F-scores for line and word segmentation, outperforming other datasets and systems. Skew correction using Hough and Affine transform was crucial for accurate line segmentation. A comparison with previous systems showed that BN-DRISHTI is a state-of-the-art Bangla handwritten recognition. To create an "End-To-End Bangla Handwritten Image Recognition system," supervised character recognition is included in this work. Skew correction before line segmentation affected the overlap accuracy between ground truth and predictions, impacting automatic evaluation results.

Raphaela Heil and Malin Nauwerck [14] establish a baseline for handwritten stenography recognition using the LION dataset. Integration of stenographic domain knowledge and pre-training techniques significantly improves recognition results. The LION dataset consists of Lindgren's drafts and other texts, encouraging future research in handwritten stenography recognition. The Melin stenographic system used in the dataset is based on phonetic symbols and abbreviations, posing a challenge for recognition. The study highlights the challenging nature of automatically transliterating handwritten stenography. The LION dataset's high error rates indicate the difficulty of stenography recognition.

George Retsinas et al. [15] achieve state-of-the-art results with a simple architecture, outperforming other methods. Proposed best practices offer practical training and performance improvement modifications using CNN-LSTM techniques. Competitive results are obtained on the IAM dataset for line-level recognition. The system's baseline network performs poorly without additional modules. The system's simplicity may limit its adaptability to more complex datasets or tasks requiring advanced features.

Ambadas Shinde Yogesh Dandawate [16] focuses on a CNN-based OCR framework that perceives handwritten Marathi words and gives great-quality printed Marathi text—created a custom training dataset due to the restricted availability of the Marathi training dataset. Nine thousand three hundred sixty words (104 words with 90 images each) make up the dataset, created with help from people between the ages of 8 and 45. Because of the system's adaptable response to various handwriting styles, it is the best option for digitizing manually written Marathi information.

Mimansha Agrawal et al. [17] analyze the approach for recognizing handwritten Devnagari characters using deep learning techniques. CNN and other Neural Networks, such as ANN and RNN, have used the five essential stages of this recognition system segmentation: prediction, feature extraction, pre-processing, and post-processing. The author has analyzed the approach used to recognize handwritten Devanagari characters. The future system has the potential for research in Devnagari word and sentence recognition and full handwritten documents with half characters.

Manoj Sonkure et al. [18] seek to determine, by taking into account several factors, including the database being utilized, sample size, training, test set ratio, class size, data normalization size, and recognition accuracy, the most accurate classification strategy for Handwritten Devnagari Script Recognition. It provides a literature review on the Devanagari script and discusses implementing different network models such as BLSTM, CNN, and a hybrid CNN-

BLSTM. The recognition accuracy of CNN is found to increase with the number of convolutional layers. The use of transfer learning for DCNN and implementing a hybrid model with KNN and ANN classifiers for recognizing individual vowels and consonants can be further investigated to enhance the recognition performance. The system can include character and text detection in scene images, expanding its applicability beyond isolated character recognition.

Sarayut Gonwirat Olarik Surinta [19] concentrates on the problem of predicting the handwritten text image sequence pattern, which is challenging since many writing styles exist, there needs to be more training data, and background noise can arise in the text images. They said that a combination of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), called CRNN, have been successful in word recognition for handwritten text. Further research could focus on exploring different variations of the CRNN architecture or other deep learning models for improvement in the accuracy and performance of handwritten recognition systems.

D. Saraswathi and Sanaa Mohamed Sherif [20] are involved in their system, capturing, recognizing, and converting characters from various sources into machine-coded forms. Convolutional neural networks (CNNs) were used in the system, which are deep learning algorithms widely used in computer vision and image classification. The system also utilized OpenCV, an open-source computer vision library, to visualize the predicted output. Data normalization was performed to improve accuracy by balancing the inputs and outputs of the model. Methods can be explored to increase the recognition rate, such as incorporating segmentation processes to recognize words, sentences, and paragraphs. The model can be extended to recognize different languages, expanding its applicability.

Evans Ehiorobo et al. [21] said offline handwriting recognition helps digitize handwritten documents, such as forms, postal addresses, and free-form random documents. The typical process of building a software application for offline handwriting recognition involves gathering and labeling a collection of images to train a neural network and then creating and training the neural network using the acquired image data. The author suggests training a Generative Adversarial Network (GAN) to automatically produce handwritten text images as a different method of gathering training data for handwritten text recognizers. The GAN is trained using the Extended MNIST dataset. Further research can address the challenges in training GANs, such as adequately minimizing loss functions, avoiding similar outputs for different input data, and ensuring effective feedback between the generative and discriminative networks.

Rohini Khalkar et al. [22] propose a Deep Learning for Handwritten Text Recognition system that aims to develop a deep learning-based application for HTR using Convolutional Neural Networks (CONVNETs) to achieve higher accuracy. The training set consists of 7850 images, and the validation set has 876 images. Pre-processing techniques like inversion, grayscale inversion, and thinning are applied to the photos, followed by segmentation using image processing techniques. Making the model more versatile and extending its use to additional languages like Hindi, Marathi, and other regional languages is one possible area for advancement. Improving the model's accuracy and efficiency is another area of emphasis.

Sonawane, P. K., & Shelke, S. [23] proposed and implemented the transfer learning mechanism with the use of the pertained CNN model, i.e., Alexnet learning, which has been trained on over a million photos and is capable of classifying images into 1000 object categories for the recognition of Devanagari characters. Rather than training a network with randomly initialized weights from the beginning, transfer learning uses the pre-trained model as a starting point to learn a new task. This method is far faster and simpler. Transfer learning works better for quicker and more effective training with fewer examples. A dataset including around 16870 samples of the 22 consonants in the Devanagari script is used to train the algorithm. The implemented model gives impressive results, with 94.49% validation accuracy and 95.46% test accuracy.

Aneja, N., & Aneja, S. [24] did the performance analysis of a pre-trained deep learning model, i.e. In terms of accuracy attained in the first epoch, the total number of epochs needed to achieve the best accuracy, the total time taken by 15 epochs, average epoch time, and the number of features required by the model, AlexNet, DenseNet, Vgg, and Inception are the best options for handwritten Devanagari character recognition. They tested models with the DHCP dataset and found that Inception performed best with an accuracy of 99%, and the DenseNet model performed worst.

Bisht, M., & Gupta, R. [25] used the Convolutional Neural Network (CNN) model with Relu and Softmax activation. Correctness metrics and the categorical cross-entropy loss function are used to gauge the model's correctness. According to the scientists, with an accuracy of 99.28% and a dropout and stochastic gradient descent (SGD) optimizer, CNN is one of the finest character and numeral recognition algorithms, which trained the model using the DHCD dataset.

Nagvanshi S. et al. [26] discussed various methods for handwritten recognition. The authors found that the Convolutional Neural Network (CNN) provides better accuracy than any other method. Methods like the Slant Correction Method were found to be less accurate for this recognition task.

Deore, S. P., & Pravin, A. [27] classified Devanagari handwritten characters using a contemporary Deep Convolutional Neural Network (DCNN) analysis and fine-tuning technique. They used a dataset of 5800 images. The authors also implemented the VGG16 deep learning model. They got 94.84 and 96.55% accuracy using the first and second approaches, respectively.

Gupta P. et al. [28] said a reliable software solution needs to recognize the traditional scripts. They used the CNN approach for Devanagari character recognition and reached an accuracy of 95.6%. According to the author's findings, it is one of the trustworthy methods for handwritten character recognition.

Chakraborty R. et al. [29] used the Convolution Neural Network (CNN) to recognize characters of Bangla and Devanagari languages. They used 1800 Devanagari characters and 10,000 Bangla primary characters for their experiments, and their accuracy rates were 99.65% and 98.87%, respectively.

S. Sawant et al. [30] proposed and implemented the CNN model with two different architectures to recognize Modi Script. Max-pooling is completely connected, and two convolution layers comprise the initial CNN model. The second architecture shall consist of three convolution and max-pooling layers and three wholly connected layers. The proposed model is tested over a dataset collected from 100 different writers with 57-character classes, giving test accuracy of 95.44% and 95.97%, respectively, on the Modi character.

Dokare I. et al. [31] used a Convolutional Neural Network (CNN) to recognize Devanagari consonants, vowels, and numerals. The authors used the Devanagari Handwritten Character Dataset (DHCD) and Nepali Handwritten Character Datasets (NHCD) for training. The authors found the recognition task to be more accurate.

Saraswathi, D., & Sherif, S. M. [32], A convolutional neural network (CNN) was constructed to train on datasets of alphabets and digits to recognize them. Furthermore, the predictions made by the trained model were visualized using OpenCV. The model offers the highest accuracy of 97.65% and 97.91% for alphabets and numerals.

S. D Pande et al. [33] configured and implemented the Convolutional Neural Network (CNN) architecture, which consists of 4 convolutional, 2 max-pooling, 3 dropouts, 1 flatten, and 2 dense layers for the recognition of Devanagari handwritten text recognition, which extracts the statistical features from isolated characters. The model was tested on the DHCD dataset, which had 46 classes of characters and a thousand different images. The result of implementation was found to be better than the existing methods of recognition.

The literature survey uses handwritten Devanagari character recognition methods, focusing on deep learning methods like Convolutional Neural Networks (CNN), CNN-SVM hybrids, Long Short-Term Memory (LSTM), and EfficientNet. There are different OCR methods analyzed by various studies, where CNN is found to be useful for feature extraction, and LSTM benefits sequence learning for compound characters. Hybrid models of CNN-SVM utilize the feature learning aspect of CNN and the strong classification performance of SVM. Attention-based CNN architectures have recently gained popularity owing to their power to pay attention to essential features to enhance recognition accuracy. Experiments have also explored data augmentation, preprocessing, and the role of transfer learning. The survey brings forth challenges like segmentation, handwriting style variations, and the scarcity of annotated datasets. In total, the review highlights the progress made at Devanagari OCR. It encourages future research in multilingual OCR and real-world applications such as document digitization, historical text preservation, and automated text processing.

The suggested attention-based CNN focuses on localized stroke-level feature re-weighting, which makes it computationally efficient and more appropriate for limited-data handwritten Devanagari compound character recognition than recent transformer-based OCR models that depend on large-scale datasets and intensive pre-training.

3. Deep Learning Algorithms

The proposed handwritten Devanagari character recognition system employs multiple deep-learning algorithms to improve accuracy and efficiency. The system leverages Convolutional Neural Networks (CNN), CNN-SVM, Long Short-Term Memory (LSTM), EfficientNet, and a proposed Attention-based CNN model. Each model is selected based on its ability to capture complex character structures, extract features effectively, and handle variations in handwritten text. The following sections describe each algorithm in detail.

3.1. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are extensively applied in image classification tasks, such as handwritten Devanagari character recognition. CNNs process various layers, including convolutional, pooling, and fully connected layers [34]. Convolutional layers extract key features from input images with the help of filters, detecting spatial hierarchies such as edges, curves, and textures. Pooling layers decrease the spatial dimensions while maintaining important information, thus increasing computational efficiency. The connected layers mapped extracted features to class labels for classification. Non-linearities are introduced by activation functions such as ReLU, which enhance learning ability. CNN models learn features from handwritten Devanagari characters efficiently and are consequently very effective for character recognition applications. Fig. 1. displays the CNN algorithm's architecture.

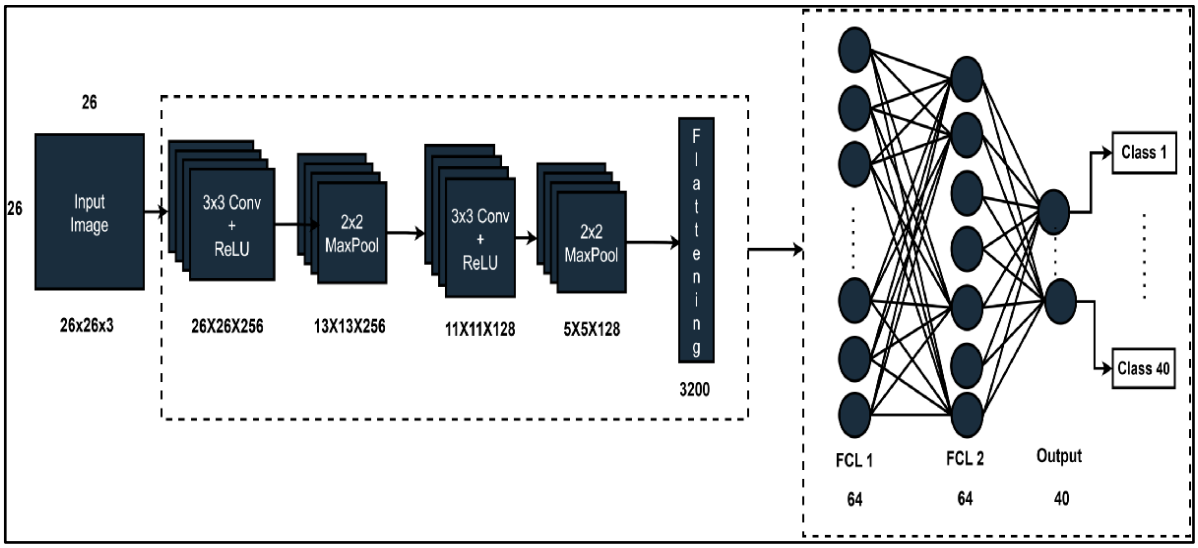


Fig. 1. Architecture of CNN algorithm [35]

- Input Layer: Accepts raw image data, typically in the form of pixel values. The input shape depends on the image resolution and number of color channels (e.g., grayscale or RGB).
- Convolutional Layer (Conv Layer): Applies filters (kernels) to detect spatial patterns such as edges, curves, and textures. The filters slide over the image, performing element-wise multiplication and summation (convolution operation), generating feature maps.
- Activation Layer (ReLU): Introduces non-linearity by converting negative values to zero, enabling the network to learn complex patterns effectively.
- Pooling Layer: Reduces the spatial dimensions of feature maps while retaining essential features, improving computational efficiency and reducing overfitting. Common types include:
    - Max Pooling: Retains the maximum value in a region.
    - Average Pooling: Computes the average value in a region.
- Flatten Layer: Converts the 2D feature maps into a 1D vector to prepare for classification in the fully connected layers.
- Fully Connected Layer (FC Layer): Connects all neurons, integrating extracted features and mapping them to output categories.
- Output Layer: Uses an activation function like Softmax (for multi-class classification) or Sigmoid (for binary classification) to predict class probabilities.

Each layer in CNN plays a crucial role in progressively extracting and refining features to improve recognition accuracy.

3.2. CNN-SVM Hybrid Model

The CNN-SVM hybrid model merges the feature extraction strength of Convolutional Neural Networks (CNN) with the strong classification strength of Support Vector Machines (SVM) [36]. CNNs are good at learning spatial hierarchies and deep feature extraction from images but are sometimes not able to achieve the best classification accuracy using traditional softmax classifiers employed in CNN. Instead, SVM, a powerful machine learning tool, is incorporated into the CNN pipeline to boost classification performance. The architecture diagram of the hybrid CNN-SVM algorithm is shown in Fig. 2.

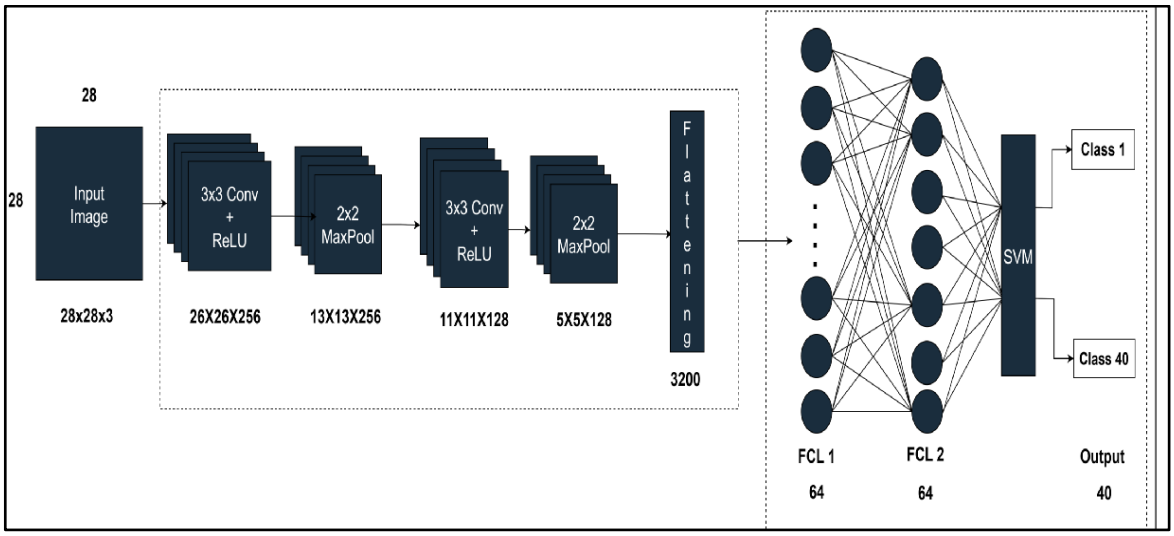


Fig. 2. Architecture of CNN-SVM algorithm

The Hybrid CNN-SVM approach represents a fusion of CNNs and Support Vector Machines (SVMs) in compound Devanagari character classification. This hybrid architecture capitalizes on the strengths of both CNNs, renowned for their feature learning capabilities in image data, and SVMs, known for their robust classification performance [37]. The architecture of layers of CNN-SVM is explained below.

- Feature Extraction with CNN: The CNN model extracts meaningful features from handwritten Devanagari character images using convolutional layers, pooling layers, and activation functions.
- Flattening & Fully Connected Layers: The extracted features are flattened into a vector representation.
- Classification using SVM: Instead of the softmax function, the output features are fed into an SVM classifier, which finds an optimal decision boundary to separate character classes effectively.

This hybrid model improves the accuracy of Devanagari character recognition by leveraging CNN's deep feature extraction and SVM's strong classification capability, making it suitable for complex handwritten character datasets.

3.3. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a Recurrent Neural Networks (RNN) class specifically built to learn temporal dependencies in sequential data. In contrast to regular RNNs, LSTMs can learn long-term dependencies because of their unique memory cell structure [38]. This part explains the architecture and training of LSTM networks for Devanagari compound character recognition. The LSTM network used in this study consists of the following layers, as shown in Fig. 3.

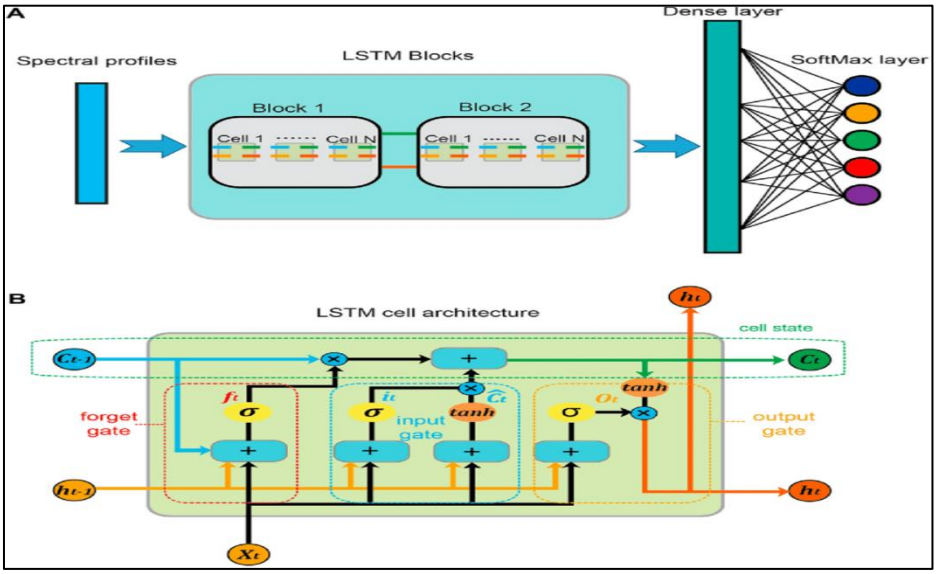


Fig. 3. Architecture diagram LSTM [39]

LSTM consists of the following key components:

- Input Gate: Determines how much new information should be stored in the memory cell.
- Forget Gate: Decides what past information should be discarded from the memory cell.
- Cell State: Stores relevant information across time steps, helping in long-term dependency learning.
- Output Gate: Controls the final output by filtering helpful information from the memory cell.

By integrating LSTM into character recognition models, the system can improve the accuracy of handwritten Devanagari script recognition, handling variations in stroke patterns and character structures effectively.

Although the input consists of isolated character images, the LSTM model is used by interpreting each character image as sequence of feature vectors extracted along the horizontal direction. The sequential representation allows the LSTM to model stroke progression and spatial dependencies within the multilevel compound characters, which is particularly beneficial for script such as Devanagari multilevel compound characters that has complex structure and overlapping strokes.

3.4. EfficientNet

EfficientNet is a family of convolutional neural networks designed to optimize the trade-off between accuracy and computational efficiency. Introduced by Google researchers, EfficientNet uses a compound scaling method to uniformly scale the network's depth, width, and resolution [39]. This approach allows EfficientNet to achieve state-of-the-art accuracy with significantly fewer parameters and FLOPs than traditional architectures like ResNet and VGG. The EfficientNet architecture uses mobile inverted bottleneck convolution (MBConv) layers and a squeeze-and-excitation (SE) mechanism. The following are the key components of the EfficientNet-B0 architecture used in this study. The architecture diagram of the EfficientNet algorithm is shown in Fig. 4.

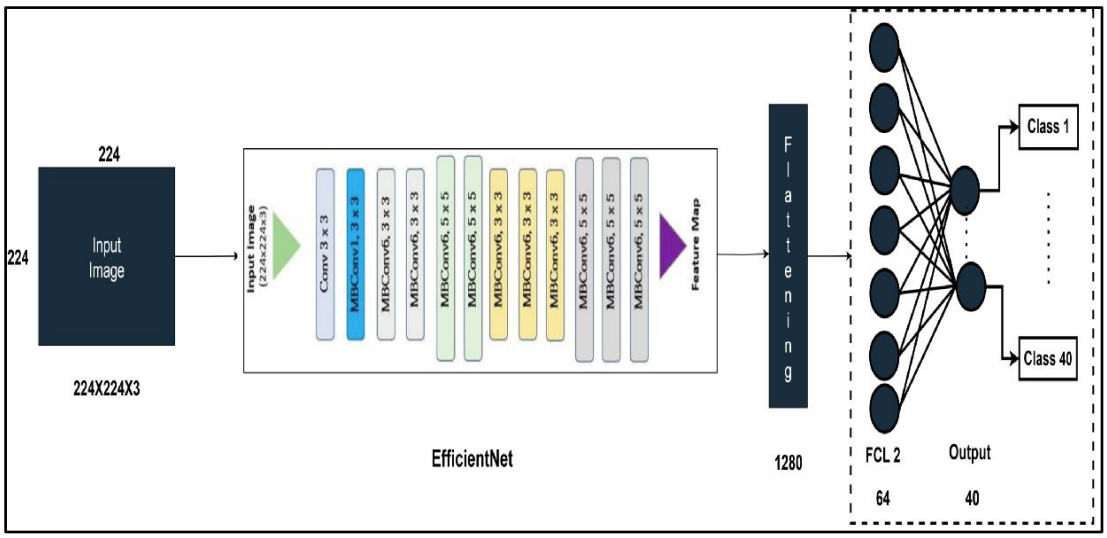


Fig. 4. Architecture diagram Efficientnet [40]

EfficientNet is built using Mobile Inverted Bottleneck Convolutions (MBConv) and Squeeze-and-Excitation (SE) blocks, which enhance feature extraction. The key components include:

- MBConv Layers: Reduce computational complexity while retaining accuracy using depthwise separable convolutions.
- Squeeze-and-Excitation (SE) Blocks: Improve model focus on essential features by dynamically re-weighting channel-wise feature maps.
- Global Average Pooling (GAP): Reduces spatial dimensions and feeds the most relevant information into fully connected layers.
- Fully Connected and Output Layers: The extracted features are passed through dense layers, and final classification is done using softmax activation.

3.5. Proposed Attention-based CNN Model

The proposed Attention-Based Convolutional Neural Network (CNN) adds an attention mechanism to make the network more focused on the most important areas of input images. The mechanism enhances the model's performance

by dynamically emphasizing key features and eliminating irrelevant or redundant information. The architecture incorporates regular convolutional layers with an attention mechanism, making it highly efficient for Devanagari compound character recognition. The architecture of the proposed Attention-Based CNN is divided into the following components and shown in Fig. 5.



Fig. 5. Architecture diagram Attention-based CNN.

The Attention-Based Convolutional Neural Network (CNN) model improves standard CNN architectures by adding an attention mechanism that dynamically weights important features and ignores irrelevant information. This enhances recognition accuracy, especially for challenging classification tasks like handwritten Devanagari compound character recognition.

The use of attention based mechanism is used instead of using the generic channel or spatial attention, which is lightweight and applied after the intermediate convolutional layers to re-weight the stoke-level feature maps. This helps to reduce the computational overhead while improving the discrimination among he similar compound characters.

The proposed architecture consists of the following key components:

- **Input Layer**: The input consists of grayscale images of size 64×64×164 \times 64 \times 164×64×1. The pixel values are normalized to the range [0,1] [0,1] [0,1] to improve numerical stability and accelerate training.
- **Convolutional Layers**: These layers extract hierarchical spatial features from the input images using a set of learnable filters (kernels). The convolution operation is mathematically expressed as:

$$f(i,j) = \sum_{k=0}^{n-1} \sum_{l=0}^{m-1} I(i+k, j+l) \cdot K(k,l) \quad (1)$$

where:
$I(i,j)$ represents the input image or feature map,
$K(k,l)$ is the kernel,
$f(i,j)$ is the resulting feature map.
A Rectified Linear Unit (ReLU) activation function is applied to introduce non-linearity:

$$f(x) = \max(0, x) \quad (2)$$

- Attention Module: The attention module enhances the model's ability to focus on essential image regions. It operates through three primary steps:
  - Global Context Extraction: Computes a global summary of each channel using global average pooling

$$g_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} f_{i.j.c} \quad (3)$$

  - Attention Score Computation: Uses a small, fully connected network with a sigmoid activation function to generate attention scores

$$\alpha_c = \sigma(W \cdot g_c + b) \tag{4}$$

- Feature Re-weighting: The attention scores are applied to the feature maps to emphasize relevant features
-

$$\hat{F}_{i,j,c} = \alpha_c . f_{i,j,c} \tag{5}$$

- Pooling Layers: Max pooling reduces spatial dimensions while retaining essential features. The max pooling operation is defined as:

$$P(i,j) = \max_{m,n,\in W} f(i+m, j+n) \tag{6}$$

- Fully Connected Layers: These layers map the extracted and refined features to class labels using:

$$y = W \cdot x + b \tag{7}$$

Where:
W is the weight matrix,
x is the input vector,
b is the bias term,
y is the output class score.

- Output Layer: The final output is determined using the softmax activation function, which converts raw class scores into probabilities:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{8}$$

where:
z_i is the raw score for class i,
k is the total number of classes.

This architecture is particularly well-suited for Devanagari compound character recognition, where characters exhibit complex ligatures, stroke variations, and multi-component structures, making conventional CNN architectures prone to misclassification. The qualitative behavior of attention mechanism can be inferred from the improved discrimination of visually similar compound characters, indicating that the model assigns higher important to structurally salient region such as ligatures and stroke intersections.

4. Methodology

This section presents the detailed methodology for recognizing handwritten Devanagari multilevel compound characters using deep learning techniques. The proposed approach involves multiple stages: dataset preparation, preprocessing, model training, and evaluation. The methodology is designed to address the challenges associated with the complex structure of Devanagari script, such as overlapping strokes, compound characters, and variations in handwriting styles. The study explores deep learning architectures, including CNN, CNN-SVM, LSTM, EfficientNet, and the proposed attention-based CNN model. The attention mechanism incorporated in the CNN model enhances feature extraction, enabling the system to focus on the most relevant regions of the input images. Additionally, data augmentation techniques are employed to improve model generalization, making it robust to variations in handwriting. The overall methodological framework ensures a comprehensive analysis of different models and their effectiveness in enhancing recognition accuracy for handwritten Devanagari characters. The block diagram of the Handwritten Devanagari Compound Character Recognition is shown in Fig. 6.

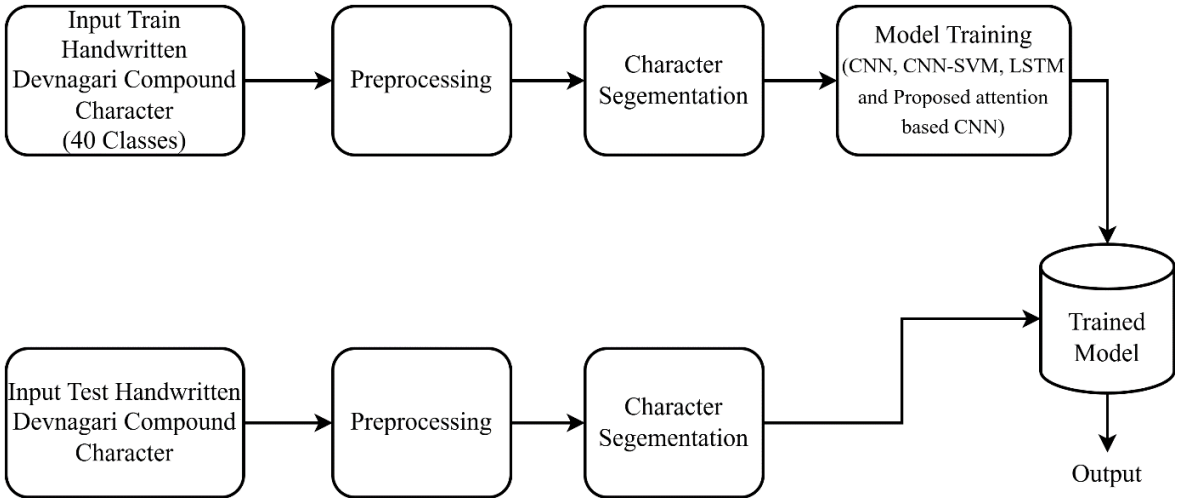


Fig. 6. Block diagram of the Handwritten Devanagari Compound Character Recognition

4.1. Dataset

The dataset used for training and evaluation consists of Marathi handwritten multilevel compound characters collected from 21 different writers across various age groups, ranging from 13 to 66 years, ensuring diversity in writing styles. Both male and female contributors participated in the dataset creation to enhance generalization. A total of 40 Marathi handwritten multilevel compound characters were considered for this study. The data collection process was meticulously designed using A4-sized sheets containing 10 rows and 7 columns, resulting in 77 cells per page. Each writer was instructed to write a single character 77 times on an entire page to maintain consistency.

For each of the 40 selected characters, a dataset of 1,617 samples per character was collected, leading to a comprehensive database of 64,680 isolated handwritten Marathi multilevel compound characters. This well-structured dataset, consisting of 40-character classes written by 21 individuals, is a robust foundation for training deep learning models, enabling efficient recognition of complex handwritten Marathi compound characters. Fig.7. shows the 40 handwritten multilevel compound character classes considered.



Fig. 7. Set of 40 handwritten multilevel compound characters

Fig. 8. depicts the Sample A4 sheets for Handwritten multilevel compound characters.

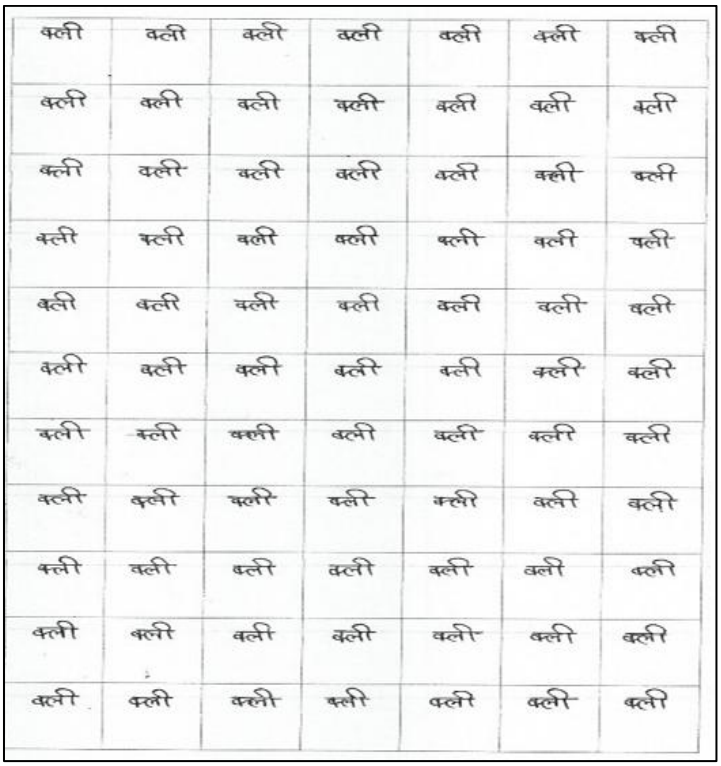


Fig. 8. Sample A4 sheets for Handwritten multilevel compound characters

The dataset consists of 64,680 Marathi handwritten multilevel compound characters collected from writers across various age groups ranging from 13 to 66 years, including both male and female contributors. This diverse dataset ensures variations in handwriting styles, stroke thickness, and character formation, which are essential for improving the generalization capability of the recognition model.

To facilitate practical training and evaluation, the dataset is split into two subsets:

- Training Set: 80% of samples were used for model learning and optimization.
- Testing Set : 20 % of samples reserved for final model evaluation on unseen data.

This structured dataset enables the deep learning model to recognize complex handwritten Marathi multilevel compound characters effectively while ensuring robust performance across different writing styles.

4.2. Preprocessing

Preprocessing and segmentation are crucial steps in handwritten character recognition, ensuring that the input images are clean, noise-free, and optimally formatted for efficient feature extraction and classification. These steps help improve the overall accuracy and robustness of the recognition system by addressing issues such as noise, distortions, and inconsistencies in handwritten samples.

The primary goals of preprocessing include:

- Enhancing the quality of input images to improve recognition performance.
- Standardizing the image format to ensure uniformity across all samples.
- Eliminating noise and distortions such as ink smudges, uneven strokes, and background artifacts.
- Preparing images for segmentation and feature extraction, ensuring characters are properly isolated for model training.

To achieve the objectives of preprocessing, several techniques are applied to enhance image quality and prepare the data for efficient feature extraction and classification. Grayscale conversion is the first step, where RGB images are transformed into grayscale to simplify computations while preserving essential character details. Next, binarization is performed using adaptive thresholding, which converts images into a black-and-white format, making distinguishing characters from the background easier. Noise removal follows, utilizing Gaussian filtering and morphological operations to eliminate unwanted artifacts and distortions in handwritten samples. To ensure uniform orientation, skew correction is applied, detecting and adjusting any rotation or tilt in the handwritten text. Additionally, normalization is conducted to standardize character size and intensity values, ensuring consistency across all dataset samples.

Once preprocessing is complete, segmentation is carried out to isolate individual characters from the input images. This involves contour-based character separation, which detects handwritten components, and connected component analysis, which identifies and extracts characters from grouped text. An essential step in the Devanagari script is Shirorekha (header line) removal, as the header line connects multiple characters and must be handled carefully to ensure accurate segmentation. These preprocessing and segmentation techniques ensure the handwritten character images are clean, well-structured, and ready for effective feature extraction and classification by deep-learning models.

Binarization is used to improve stroke clarity and reduce the background noise, Shirorekha elimination stops structural dominance and character merging, segmentation guarantees correct isolation of compound characters. these processes enhance overall recognition accuracy.

4.3. Model Training using the DL algorithms

The training process is crucial in developing a robust handwritten Devanagari multilevel compound character recognition system. Deep learning algorithms are employed to learn and extract meaningful patterns from the dataset, enabling accurate classification of characters. The proposed study leverages multiple deep learning architectures, including Convolutional Neural Networks (CNN), CNN-SVM, Long Short-Term Memory (LSTM), EfficientNet, and the Attention-based CNN model to improve recognition performance. The model training begins with data preparation, where the dataset is divided into training (80%) and testing (20%) subsets. The training set is used to train the model, while the testing set evaluates its performance on unseen data. The images undergo preprocessing, including grayscale conversion, noise removal, and normalization, ensuring uniformity in input samples.

During training, CNN-based architectures extract hierarchical features from character images, where convolutional layers identify essential patterns such as edges, curves, and textures. Activation functions like ReLU (Rectified Linear Unit) introduce non-linearity, improving the model's ability to learn complex character structures. Pooling layers reduce dimensionality while retaining critical features, optimizing computational efficiency. Fully connected layers map extracted features to output labels, and a softmax activation function assigns class probabilities.

For hybrid CNN-SVM models, the CNN extracts deep features, which are then classified using Support Vector Machines (SVMs) instead of a softmax classifier, improving robustness in character classification. LSTM-based models, on the other hand, are integrated to handle sequential dependencies in handwriting patterns and learn long-term dependencies between strokes in compound characters.

The EfficientNet model optimizes performance using compound scaling and adjusting depth, width, and resolution to achieve state-of-the-art accuracy with minimal computational cost. Finally, the Attention-based CNN model further enhances character recognition by incorporating an attention mechanism, enabling the network to focus on key image regions while suppressing irrelevant information. This improves feature extraction, leading to better classification accuracy.

The Attention-Based CNN Model further enhances character recognition by incorporating an attention mechanism, which dynamically focuses on the most critical regions of the image. This mechanism works by:

- Extracting global context from each feature map using global average pooling.
- Computing attention scores through a small, fully connected network with a sigmoid activation function.
- Feature re-weighting, where attention scores are applied to feature maps, emphasizing important features and suppressing irrelevant information.

This attention mechanism allows the model to prioritize highly discriminative features, improving classification accuracy. The attention-enhanced feature maps are then passed through additional convolutional and pooling layers, followed by fully connected layers for final classification.

4.4. Model Evaluation

Precision, Recall, F1 Score, and Accuracy are the four main assessment measures used to assess the performance of the suggested strategy. These metrics offer a thorough evaluation of the model's functionality.

- Precision: It describes how well the model predicts positive values by finding the proportion of actual positive cases to all predicted positive instances.

$$Precision\ (P) = \frac{TP}{Tp+FP} \tag{9}$$

- Recall: By dividing the number of correctly identified positive cases by the total number of positive instances, Recall, also known as Sensitivity or the True Positive Rate—calculates the model's capacity to identify positive cases.

$$Recall\ (R) = \frac{TP}{Tp+FN} \tag{10}$$

- F1 Score: The F1 Score is a well-balanced statistic that considers both Precision and Recall and is particularly beneficial in imbalanced datasets. It is defined as the harmonic mean of Precision and Recall

$$F1 - Score = 2 * \frac{P \times R}{P + R} \tag{11}$$

- Accuracy: By calculating the proportion of correctly categorized instances to all cases, accuracy assesses the model's overall correctness.

$$Accuracy = \frac{TP+TN}{Tp+TN+FP+FN} \tag{12}$$

- Statistical Significance Analysis: To check whether the performance difference between the models are statistically meaningful, a statistical significance analysis Is conducted. All models ate trained and evaluated using different random seeds. The mean and standard deviation of validation accuracy were computed to quantify the performance variability across the run. The 95% confidence interval (CI) were calculated to estimate the reliability of accuracy values.

5. Result and Discussion

The performance of various deep learning algorithms for handwritten compound Devanagari character recognition is analyzed in this section. Multiple architectures, including CNN, CNN-SVM, LSTM, EfficientNet, and the proposed Attention-based CNN model, were evaluated based on their training accuracy, validation accuracy, training loss, validation loss, and execution time. The results are examined to determine the most effective model for recognizing complex Devanagari characters.

Hardware Used: Dell Precision 3640 workstation equipped with an Intel Core i9-10900 CPU (10 cores, up to 5.2 GHz), 128 GB DDR4 RAM, and an NVIDIA Quadro RTX 5000 GPU with 16 GB memory.

A. Results of CNN for handwritten compound Devanagari character recognition

This sub-section discusses the outcome of CNN-based handwritten compound Devanagari character recognition. The CNN model was trained with various combinations of epochs and optimizers to test its performance in recognizing intricate character compositions. The effect of these differences on training accuracy, validation accuracy, training loss, and validation loss was investigated. The outcomes for various epochs and optimizers are presented in Table 1, which reveals the model's performance under various training scenarios.

Table 1. Performance analysis of CNN based on Epochs and Optimizer

No. of Epochs	Optimizer	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Execution Time
50	Adam	61.76	1.1701	75.71	0.7915	10645.91
50	RMSProp	71.41	0.9618	83.37	0.5700	10848.89
40	Adam	96.09	0.0791	87.48	0.5919	4960.18
40	RMSProp	95.22	0.1186	83.65	0.7429	9472.15
30	Adam	95.75	0.0866	89.04	0.4776	7664.07
30	RMSProp	95.87	0.0925	88.38	0.5160	3948.50
20	Adam	92.61	0.1793	84.50	0.7353	22297.00
20	RMSProp	93.35	0.1723	84.73	0.6775	2638.13
10	Adam	92.68	0.1832	87.12	0.4679	1236.99
10	RMSProp	91.65	0.2323	83.29	0.5883	1321.61

The CNN model was optimized using Adam and RMSProp optimizers with various epoch settings. The report reflects that the validation accuracy is enhanced by raising epochs, but more epochs contribute to more excellent execution time. The highest validation accuracy of CNN in the case of using the Adam optimizer was 89.04% for 30 epochs, and with RMSProp, it was 88.38% for the same level of epochs. This means that the hyperparameters need to be carefully optimized in the CNN models to improve the accuracy and efficiency of execution. The accuracy and loss graph for CNN algorithm with Adam optimizer for 30 epoch is present in Fig. 9.

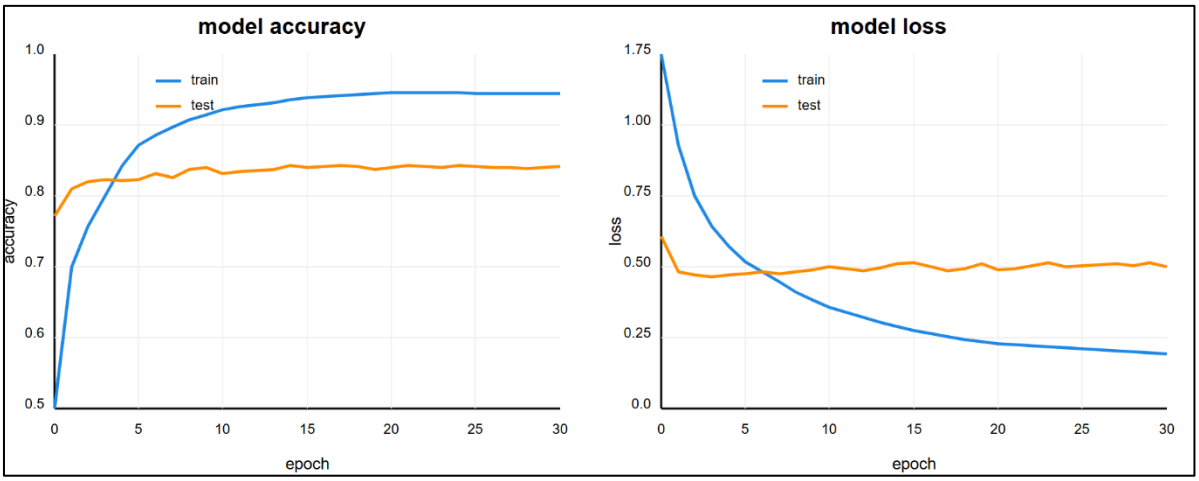


Fig. 9. Performance of CNN model for with (i) Accuracy for 30 epoch and Adam optimizer (j) Loss for 30 epoch and Adam optimizer

B. Results of CNN-SVM for handwritten compound Devanagari character recognition

This subsection discussed the performance results for the CNN-SVM hybrid model in handwritten compound Devanagari character recognition. This method merges the feature extraction power of CNN with the classification power of Support Vector Machines (SVM) to improve recognition accuracy. The model was trained on various configurations, and its performance was measured based on training accuracy, validation accuracy, training loss, and validation loss. The results for various training environments are presented in Table 2, giving a comparative evaluation of the CNN-SVM model's performance in identifying intricate Devanagari characters.

Table 2. Performance analysis of CNN-SVM based on Epochs and Optimizer

Number of Epochs	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Execution Time
50	98.60%	0.9769	97.82%	97.75%	6214.17

The hybrid model CNN-SVM was tested at 50 epochs with a training accuracy of 98.60% and a validation accuracy of 97.82%. Although CNN-SVM has improved generalization compared to the basic CNN model, it is at a more significant computational cost, as observed in the run time of 6214.17 seconds. This hybrid method shows the superiority of SVM as a classifier over softmax in classifying complex handwritten characters. The training performance graphs of CNN0SVM algorithm are shown in Fig. 10.

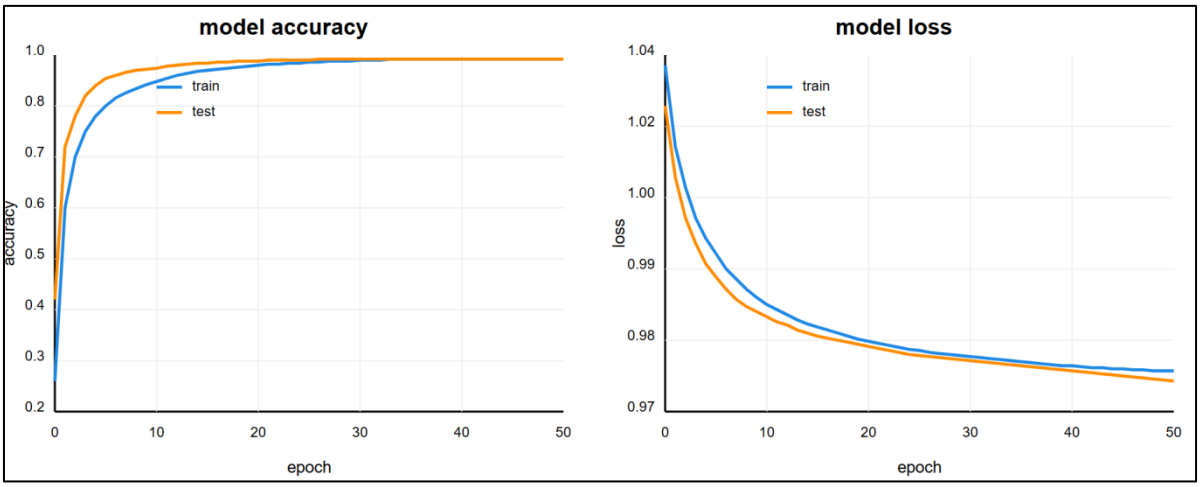


Fig. 10. Performance of CNN-SVM model for with (i) Accuracy for 50 epoch and Adam optimizer (j) Loss for 50 epoch and Adam optimizer

C. Results of LSTM for handwritten compound Devanagari character recognition

This sub-section discusses the findings of the Long Short-Term Memory (LSTM) model for handwritten compound Devanagari character recognition. LSTM is a form of recurrent neural network (RNN) best suited to learning sequential dependencies and, thus, is ideal for recognizing handwriting differences and intricate character compositions. The model was trained using various layer settings to study its effect on recognition accuracy and loss. The outcome,

such as training accuracy, validation accuracy, training loss, and validation loss, is given in Table 3, with a comprehensive performance analysis of the LSTM model for handwritten Devanagari character recognition.

Table 3. Performance analysis of LSTM based on Epochs and Optimizer

Architecture	No. of Neurons	Dropout	Train Accuracy	Train Loss	Val Accuracy	Val Loss	Execution Time
LSTM 1 Layer	128	0.2	90.26	0.2623	85.47	0.5076	1108.13
LSTM 2 Layer	128-64	0.2	91.81	0.2082	88.02	0.3953	1146.18
LSTM 3 Layer	128-64-128	0.2	94.17	0.1388	90.94	0.2713	1118.99
LSTM 4 Layer	258-128-64-32	0.2	92.71	0.1877	89.32	0.3999	1147.89
LSTM 5 Layer	258-128-128-64-32	0.2	89.23	0.3016	84.00	0.5629	1195.36

LSTM networks were experimented with various layer sizes, specifically for learning sequential dependencies of handwritten characters. The best accuracy of 90.94% with a validation loss of 0.2713 was obtained using the LSTM-3 layer model (128-64-128 neurons), beating deeper models. Further, the increase in layers above three resulted in reduced accuracy because of overfitting and increased execution time. The training and validation performance of LSTM with 3 layers (128-64-128) is shown in Fig. 11.

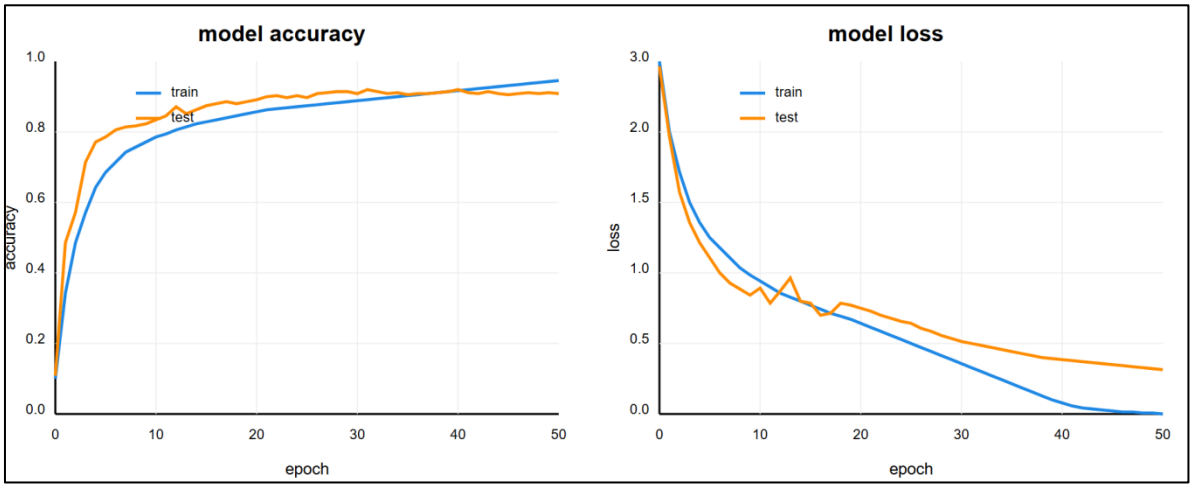


Fig. 11. Performance of LSTM model for 3 layer architecture (128-64-128) with (i) Accuracy (j) Loss

D. Results of Efficientnet for handwritten compound Devanagari character recognition

Here, the results of the EfficientNet model for handwritten compound Devanagari character recognition are discussed. EfficientNet is popular for its structured architecture, where accuracy and computation are balanced through scaling network depth, width, and resolution. The model was trained with the Adam optimizer, and various input sizes were used to study their effect on the recognition performance. The outcomes, such as training accuracy, validation accuracy, training loss, and validation loss, are tabulated in Table 4 and reflect upon the efficacy of EfficientNet in identifying intricate handwritten Devanagari characters.

Table 4. Performance analysis of Efficientnet based on Epochs and Optimizer

Input Size	Optimizer	Train Accuracy	Train Loss	Val Accuracy	Val Loss
(224,224,3)	Adam	92.61%	1.1921e-07	96.32%	1.1921e-07
(240,240,3)	Adam	90.53%	1.1921e-07	91.35%	1.1921e-07
(260,260,3)	Adam	73.60%	1.1921e-07	86.33%	1.1921e-07
(300,300,3)	Adam	89.23%	1.1921e-07	95.44%	1.1921e-07

The EfficientNet model was experimented with different input sizes (B0 to B3 configurations). EfficientNetB0 performed the best with 96.32% validation accuracy, whereas EfficientNetB2 had a lower validation accuracy of 86.33%, suggesting that scaling up after a certain point might not always be optimal. The model complexity vs. accuracy trade-off must be considered for handwritten character recognition tasks. The training and validation performance of EfficientNet with input size of 224x224x224 is shown in Fig. 12.

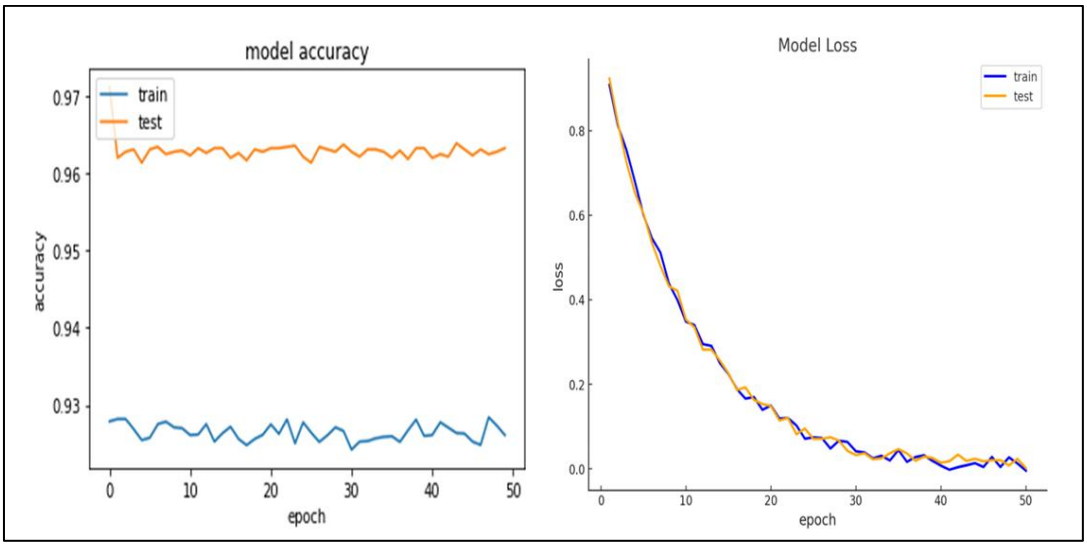


Fig. 12. Performance of Efficientnet for Adam optimizer with (i) Accuracy (j) Loss

E. Results of Proposed attention-based CNN for Handwritten Compound Devanagari Character Recognition

This subsection describes the performance of the suggested Attention-Based CNN model for handwritten compound Devanagari character recognition. It utilizes the mechanism of attention to reinforce feature extraction and can concentrate on the most informative parts of characters and down-weight the effect of less important features. The attention mechanism enhances recognition accuracy by dynamically weighting important regions of the input image, and it is especially effective for intricate character structures. To test its performance, the model was trained with various epoch configurations and optimization methods. The results, such as training accuracy, validation accuracy, training loss, and validation loss, are listed in Table 5, showing the proposed method's superior recognition ability.

Table 5. Performance analysis of Proposed attention-based CNN based on Epochs and Optimizer

Sr. No	No. of Epochs	Optimizer	Train Accuracy	Train Loss	Val Accuracy	Val Loss	Execution Time (s)
1	10	RmsProp	66.68%	0.9778	68.53%	0.9046	391.75
2	10	Adam	71.68%	0.7987	73.17%	0.7305	413.94
3	20	RmsProp	82.69%	0.4850	87.82%	0.3645	802.22
4	20	Adam	86.86%	0.3613	85.88%	0.4271	810.59
5	30	RmsProp	89.74%	0.2905	93.75%	0.2001	1201.96
6	30	Adam	90.93%	0.2307	94.07%	0.1866	1211.83
7	40	RmsProp	91.89%	0.2168	93.21%	0.1877	1600.28
8	40	Adam	92.14%	0.2015	95.47%	0.1533	1610.94
9	50	RmsProp	93.50%	0.1565	93.75%	0.2062	2038.26
10	50	Adam	93.36%	0.1596	96.66%	0.1142	2055.59

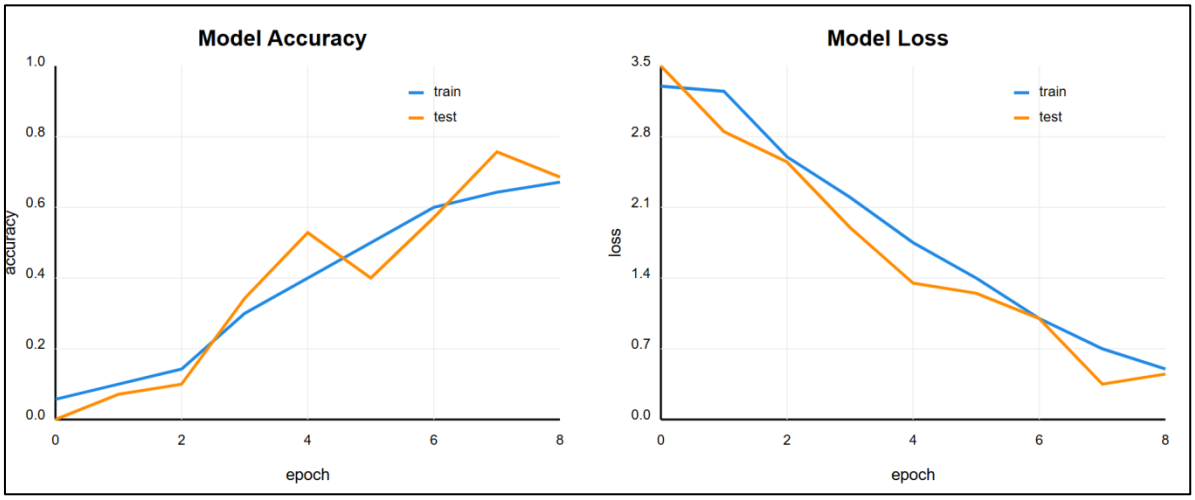


Fig. 13. Performance of attention-based CNN for 50 epochs with (i) Accuracy (j) Loss

The proposed Attention-based CNN model performed better than other architectures, reaching a validation accuracy of 96.66% with the Adam optimizer at 50 epochs and a validation loss of 0.1142. The attention mechanism improves feature extraction, enabling the model to concentrate on key areas of characters, enhancing recognition accuracy while ensuring computational efficiency. The running time was 2055.59 seconds, which, although greater than CNN, is warranted by the improved accuracy. The accuracy and loss graph of training and validation for the attention-based CNN algorithm for 50 epoch using Adam optimizer is present in Fig. 13.

The comparative analysis of the handwritten Devanagari compound character recognition is tabulated in Table 6.

Table 6. Comparative analysis of different algorithms for handwritten Devanagari compound character recognition

Algorithm	**Attributes**	**Training accuracy (%)**	**Training Loss (%)**	**Validation Accuracy (%)**	**Validation Loss (%)**
CNN	Epochs =30 Optimizer =Adam	95.75	0.0866	89.04	0.4776
CNN-SVM	Epochs =30 Optimizer =Adam	98.60	0.9769	97.82	0.9775
LSTM	Layers = 128-64-128	94.17	0.1388	90.94	0.2713
Efficient net	Optimizer=Adam	92.61	1.1921e-07	96.32	1.1921e-07
Attention-based CNN	Epochs =50 Optimizer =Adam	93.36%	0.1596	96.66	0.1142

Table 6 compares the various deep learning algorithms for handwritten Devanagari character recognition based on performance indicators such as training accuracy, training loss, validation accuracy, and validation loss. The CNN model with Adam optimizer for 30 epochs has obtained a training accuracy of 95.75% and a validation accuracy of 89.04% with a comparatively low training loss of 0.0866 and validation loss of 0.4776. The hybrid model CNN-SVM, likewise trained for 30 epochs with Adam, was the best-train accuracy-wise at 98.60% and validation accuracy at 97.82%, although it came with much more extensive training and validation losses (0.9769 and 97.75%, respectively), pointing to potential overfitting.

The validation loss reported for the CNN-SVM model corresponds to the hinge-loss objective used by SVM classifier and is therefore not directly comparable to cross-entropy losses reported for softmax-based CNN model. As a result, magnitude of loss value may appear numerically larger despite high classification accuracy.

The LSTM model, configured with 128-64-128 layers, had a training accuracy of 94.17%, validation accuracy of 90.94%, and losses of 0.1388 and 0.2713, respectively, reflecting its success in learning sequential dependencies. The EfficientNet model, optimized with Adam, had a robust validation accuracy of 96.32% with an incredibly low training and validation loss (1.1921e-07) in evidence, reflecting its effectiveness in learning with fewer parameters.

The loss values for EfficientNet is externally very small and identical across multiple configuration due to use of numerical underflow-prone loss scaling and rounding during logging. The EfficientNet implementation employed a highly confident softmax output combined with categorical cross-entropy loss, resulting in loss values approaching zero. Lastly, the Attention-based CNN, which was trained using Adam for 50 epochs, had the highest validation accuracy of 96.66% and lowest validation loss of 0.1142 among most models, which means its attention mechanism contributes significantly to feature extraction, hence the most potent model among those under comparison.

Table 7. Statistical Performance comparison

Model	**Mean Validation Accuracy (%)**	**Standard Deviation**	**95%Confidance Interval**
CNN	89.02	±0.71	±0.62
CNN-SVM	97.80	±0.44	±0.39
LSTM	90.91	±0.65	±0.57
Efficient net	96.31	±0.48	±0.41
Attention-based CNN	96.66	±0.39	±0.35

Table 7 present the statistical performance comparison of all evaluated models. The proposed attention based CNN model demonstrate the lowest standard deviation (±0.39) and the 95% confidence interval (±0.35) indicate that model is stable and has consistent performance. Although CNN_SVM has higher mean accuracy, its larger variance suggests reduced robustness.

Figure 14 presents a qualitative analysis of the proposed Attention-Based CNN system for handwritten compound Devanagari character recognition. This figure visually demonstrates how the proposed model effectively identifies and differentiates complex handwritten characters in a heterogeneous dataset. The figure includes samples that showcase the model's capability to focus on the most salient and discriminative features of handwritten input, even in the presence of challenges such as overlapping strokes, ligatures, and varying writing styles.



Fig. 14. Qualitative analysis of the Proposed System

The qualitative results in Figure 4.7 underline the strength of the attention-based approach, particularly in scenarios where traditional CNNs may fail owing to the complexity of the input or ambiguity between visually similar classes.

The Qualitative result suggest that proposed attention-based CNN implicitly emphasizes critical characters' regions, as evidenced by improved recognition of overlapping strokes and complex ligatures compared to baseline CNN model.

6. Conclusion

This study presents a deep learning-based approach for handwritten compound Devanagari character recognition, leveraging multiple architectures, including CNN, CNN-SVM, LSTM, EfficientNet, and an Attention-based CNN model. The evaluation of these models highlights the strengths and limitations of each approach. The Attention-based CNN model demonstrated superior performance, achieving a validation accuracy of 96.66%, making it the optimal choice for recognizing complex compound characters in the Devanagari script. Incorporating attention mechanisms enabled the model to focus on essential character regions, reducing misclassification and improving overall recognition accuracy. Additionally, the results indicate that CNN-SVM offers high accuracy but is computationally expensive, while EfficientNetB0 balances accuracy and execution time. This research advances handwritten character recognition systems, which can be integrated into automated document processing, OCR applications, and language translation systems for Indian scripts.

While the proposed model has achieved promising results, several enhancements can be explored to improve performance further. One potential direction is the integration of transformer-based architectures such as Vision Transformers (ViTs) or hybrid CNN-Transformer models, which have shown significant advancements in computer vision tasks. Expanding the dataset with more diverse handwriting styles and real-world noisy samples can enhance model generalization. Future research can also focus on developing lightweight and optimized models for mobile and embedded applications, enabling real-time Devanagari character recognition on low-power devices. Furthermore, incorporating multilingual character recognition capabilities by extending the model to other Indian scripts, such as Modi, Kannada, and Bengali, could significantly broaden its applicability. Finally, using self-supervised learning and generative adversarial networks (GANs) for data augmentation could help improve recognition performance for low-resource datasets.

The present work is limited to isolated character recognition. Extending the proposed attention based CNN framework to word and line level recognition using sequential modelling and segmentation is important direction for future research toward fully deployable OCR system. Future work will include the explicit visualization of attention maps to provide direct interpretability of learned attention weights and to further validate that the model focuses on meaningful character regions.

References

[1] L. Schomaker, "Lifelong Learning for Text Retrieval and Recognition in Historical Handwritten Document Collections," Series in Machine Perception and Artificial Intelligence, WORLD SCIENTIFIC, pp. 221-248, Nov. 2020. doi: 10.1142/9789811203244_0012.

[2] V. Bansal and R. M. K. Sinha, "A complete OCR for printed Hindi text in Devanagari script," Proceedings of the Sixth International Conference on Document Analysis and Recognition, Seattle, WA, USA, 2001, pp. 800-804, doi: 10.1109/ICDAR.2001.953898.
[3] R. Thatte, "Handwritten Devanagari Character Recognition Using Deep Learning," rScroll, Oct. 24, 2022. doi: 10.47611/harp.228.
[4] D. Brezina, "Context-sensitive study of the visual similarity of characters in Cyrillic, Devanagari, and Latin scripts: effects of typeface design and expertise," Center for Open Science, Oct. 18, 2024. [Online]. Available: http://dx.doi.org/10.31234/osf.io/cfgdp.
[5] H. P. P. Win, P. T. T. Khine, and K. N. N. Tun, "Converting Myanmar printed document image into machine understandable text format," 2011 Sixth International Conference on Digital Information Management, Melbourne, VIC, Australia, Sep. 2011, pp. 96-101, doi: 10.1109/icdim.2011.6093371.
[6] A. Kulkarni and A. Shivananda, Deep Learning for NLP, in Natural Language Processing Recipes, Apress, 2021, pp. 213–262. doi: 10.1007/978-1-4842-7351-7_6.
[7] C. Bhole, "Devanagari Handwritten Marathi Compound Character Recognition System," Journal of Computer Based Parallel Programming, vol. 6, no. 3, pp. 1-8, Nov. 2021, doi: 10.46610/jocpp.2021.v06i03.001.
[8] A. Srivastav and N. Sahu, "Segmentation of Devanagari Handwritten Characters," International Journal of Computer Applications, vol. 142, no. 14, pp. 15-18, May 2016, doi: 10.5120/ijca2016909994.
[9] M. B. Sahin, D. M. Yalcinkaya, R. Dharmakumar, A. Hashemi, and B. Sharif, "Retrospective phase-map synthesis for CMR datasets from magnitude-only DICOM images enabled by A.I. generative models to create large training datasets for deep learning-based image reconstruction," J. Cardiovasc. Magn. Reson., vol. 26, p. 100987, 2024, doi: 10.1016/j.jocmr.2024.100987.
[10] S. S. Gaikwad, S. L. Nalbalwar, and A. B. Nandgaonkar, “Devanagari handwritten characters recognition using DCT, geometric and hue moments feature extraction techniques,” Sādhanā, vol. 47, no. 3, 2022. doi: 10.1007/s12046-022-01872-9.
[11] O. V. Astaf'eva, "A study into the governance specifics of educational institutions through the lens of the digitization effect on the educational process," National Interests: Priorities and Security, vol. 15, no. 4, pp. 772–783, Apr. 2019, doi: 10.24891/ni.15.4.772.
[12] K. Minatani, "The impact of presentation by voice media on reading speed when visually impaired persons read complex text including high-level content," Assistive Technology Research Series, vol. 34, pp. 659–664, 2013. doi: 10.3233/978-1-61499-304-9-659.
[13] Jubaer, S. M., Tabassum, N., Rahman, M. A., & Islam, M. K. (2023). Bn-drishti: Bangla document recognition through instance-level segmentation of handwritten text images. arXiv preprint, arXiv:2306.09351.
[14] Heil, R., & Nauwerck, M. (2023). Handwritten stenography recognition and the lion dataset. Springer Nature, arXiv:2308.07799v1.
[15] Retsinas, G., Sfikas, G., Gatos, B., & Nikou, C. (2024). Best practices for a handwritten text recognition system. arXiv preprint, arXiv:2404.11339.
[16] Shinde, A., & Dandawate, Y. (2020). Convolutional neural network-based handwritten Marathi text recognition. Journal of Xidian University, 14(8), 1585–1594.
[17] Agrawal, M., Chauhan, B., & Agrawal, T. (2022). Machine learning algorithms for handwritten Devanagari character recognition: A systematic review. Journal of Science and Technology, 7(1), 1–16.
[18] Sonkure, M., Gupta, R., & Moghe, A. (2020, October). An efficient approach for handwritten Devanagari script recognition. EasyChair Preprint, No. 4370.
[19] Gonwirat, S., & Surinta, O. (2022). Cycle augment: Efficient data augmentation strategy for handwritten text recognition in historical document images. Engineering and Applied Science Research. Retrieved from https://www.tcithaijo.org/index.php/easr/index.
[20] Saraswathi, D. D., & Sherif, S. M. (2021). Handwritten text recognition system using machine learning. Kristu Jayanti Journal of Computational Sciences, 1.
[21] Ehiorobo, E., Koleoso, R., & Uwadia, C. (2020). Training of offline handwritten text recognizers using computer-generated text. Elsevier Ltd.
[22] Khalkar, R., Dikhit, A. S., Goel, A., Gupta, M., & Patil, S. (2021). Deep learning for handwritten text recognition (ConvNet & RNN). Turkish Online Journal of Qualitative Inquiry (TOJQI), 12(8), 1874–1890.
[23] Sonawane, P., & Shelke, S. (2018). Handwritten Devanagari character classification using deep learning. 2018 International Conference on Information, Communication, Engineering and Technology (ICICET), 1–4.
[24] Aneja, N., & Aneja, S. (2019). Transfer learning using CNN for handwritten Devanagari character recognition. 2019 1st International Conference on Advances in Information Technology (ICAIT), Chikmagalur, India, 293–296.
[25] Bisht, M., & Gupta, R. (2020). Multiclass recognition of offline handwritten Devanagari characters using CNN. International Journal of Mathematical, Engineering and Management Sciences, 5(6), 1429–1439. Retrieved from https://doi.org/10.33889/IJMEMS.2020.5.6.106.
[26] Nagvanshi, S., Singh, A., Yadav, S., & Deepanshi. (2020). A detailed review of different handwriting recognition methods. International Research Journal of Engineering Technology, 7(3), 3462–3467.
[27] Deore, S., & Albert, P. (2020). Devanagari handwritten character recognition using fine-tuned deep convolutional neural network on trivial dataset. Sādhanā, 45, 243.
[28] P. G. et al. (2020). Convolutional neural network-based handwritten Devanagari character recognition. 2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE). IEEE.
[29] Chakraborty, R., Mukherjee, D., Bhattacharyya, A., Mukherjee, H., Sur, M. K., Sen, S., & Roy, K. (2020). Online handwritten Bangla and Devanagari character recognition by using CNN: A deep learning concept. 68-71.
[30] Sawant, S., Sharma, A., Suvarna, G., Tanna, T., & Kulkarni, S. (2020). Word transcription of Modi script to Devanagari using deep neural network. 2020 3rd International Conference on Communication Systems, Computing and IT Applications (CSCITA 2020) - Proceedings.

[31] Dokare, I., Gadge, S., Kharde, K., Bhere, S., & Jadhav, R. (2021). Recognition of handwritten Devanagari character using convolutional neural network. 353-359.
[32] Saraswathi, D., & Sherif, S. M. (2021). Handwritten text recognition system using machine learning. Kristu Jayanti Journal of Computational Sciences (KJCS, 1(1), 58–69. Retrieved from https://doi.org/10.59176/kjcs.v1i1.2180.
[33] Pande, S., Jadhav, P., Joshi, R., Muddebihalkar, V., Rathod, S., Gurav, M., & Das, S. (2021). Digitization of handwritten Devanagari text using CNN transfer learning – A better customer service support. Neuroscience Informatics, 2, 100016.
[34] L. Li, "Convolutional Neural Networks (CNNs)-Based for Medical Image Analysis," in Proceedings of the 1st International Conference on Engineering Management, Information Technology and Intelligence, SCITEPRESS - Science and Technology Publications, 2024, pp. 546-552. doi: 10.5220/0012958600004508.
[35] P. Verma, R. Srivastava, and S. K. Tripathy, "An assessment towards 2D and 3D human pose estimation and its applications to activity recognition: A review," SN Computer Science, vol. 6, no. 190, 2025. DOI: 10.1007/s42979-025-03708-9.
[36] U. Asgher, K. Khalil, Y. Ayaz, R. Ahmad, and M. J. Khan, "Classification of Mental Workload (MWL) using Support Vector Machines (SVM) and Convolutional Neural Networks (CNN)," in Proc. 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sindh, Pakistan, Jan. 2020, pp. 1-6, doi: 10.1109/icomet48670.2020.9073799.
[37] T. Joachims, A Statistical Learning Model of Text Classification for SVMs, in Learning to Classify Text Using Support Vector Machines. Springer US, 2002, pp. 45-74. doi: 10.1007/978-1-4615-0907-3_4.
[38] F. M. Salem, Gated RNN: The Long Short-Term Memory (LSTM) RNN, in Recurrent Neural Networks. Springer International Publishing, 2021, pp. 71–82. doi: 10.1007/978-3-030-89929-5_4.
[39] Kang, Rui & Park, Bosoon & Ouyang, Qin & Ren, Ni. (2021). Rapid identification of foodborne bacteria with hyperspectral microscopic imaging and artificial intelligent classification algorithms. Food Control. 130. 108379. 10.1016/j.foodcont.2021.108379.
[40] C.-Y. Zhu, Y.-K. Wang, H.-P. Chen, K.-L. Gao, C. Shu, J.-C. Wang, L.-F. Yan, Y.-G. Yang, F.-Y. Xie, and L. Jie, "A deep learning-based framework for diagnosing multiple skin diseases in a clinical environment," Frontiers in Medicine, vol. 8, Apr. 2021. doi: 10.3389/fmed.2021.626369.

Authors' Profiles



Mrs. Ashwini Patil, research scholar at RKDF University, Bhopal, MP, India. She has 21 years of academic experience. She has life membership of ISTE. Her area of interest is Machine Learning, Networking, and Programming. She has published 35 research papers in National/International Journals and Conferences. She has published book entitled "Commence Web Development with PHP and MySQL" in 2014-15 and also published an e-book entitled "C Programming Language: Journey from Beginner to Master" on Amazon Kindle in academic year 2024-25.



Dr. Puneet Dwivedi working as an Associate Professor in the Department of Computer Science and Engineering, RKDF College of Technology & Research, RKDF University, Bhopal, has about 9 years of teaching experience. He has life membership of ISTE. He completed Bachelor of Engineering in Computer Science & Engineering branch in 2008 and Master of Technology in Software Engineering branch in 2016 from Rajiv Gandhi Prodyogiki Vishwavidyalaya Bhopal. He was awarded Doctor of Philosophy in 2021. He has published book entitled "Python Programming in Data Science". He has Published 17 research papers in refereed International and National journals and he has 5 patents. His areas of research include Deep Learning, Internet of Things (IoT), Blockchain Technology, Cyber Security, Network Security.

How to cite this paper: Ashwini B. Patil, Puneet Dwivedi, "Deep Learning-Based Recognition of Handwritten Devanagari Multilevel Compound Characters: A Performance Comparison", International Journal of Engineering and Manufacturing (IJEM), Vol.16, No.2, pp.111-131, 2026. DOI:10.5815/ijem.2026.02.07