Received: 20 Nov 2025, Accepted: 30 Dec 2025, Published: 04 Jan 2026
Digital Object Identifier: https://doi.org/10.63503/j.ijcma.2025.189

Research Article

Digitally Preserving the Devnagri Script

Preeti Bala Tripathi[1], Neeru Saxena[2]* , Shivansh Chauhan[3],Amaan Asad[4], Surabhya Chandra[5], Sidhartha Verma[6]

[1, 2, 3, 4, 5, 6] IMS, Ghaziabad (University Course Campus)

Preeti.bala@imsuc.ac.in[1], neeru.saxena@imsuc.ac.in[2], shivanshchauhan.bca2023@imsuc.ac.in[3], amaanasad.bca2023@imsuc.ac.in[4], siddharthverma.bca2023@imsuc.ac.in[5], surabhyachandra.bca2023@imsuc.ac.in[6]

*Corresponding author: Neeru Saxena, neeru.saxena@imsuc.ac.in

ABSTRACT

Devanagari is one of the most widely used and ancient notation systems in South Asia, primarily used for languages like Hindi, Sanskrit, Marathi, and Nepali. Devanagari is used to write ancient languages of the Hindu Holy Writ, philosophy, and classical literature. It preserves the centuries of India's cultural, religious, and it's intellectual heritage. Some handbooks like the Bhagavad Gita, Vedas, and Upanishads are also available in Devanagari. Therefore, it's our responsibility that we bring Devanagari in digital form. This paper presents a new system to digitize the handwritten Images based on image processing methods along with Deep Convolutional Neural Network (DCNN). Handwritten character recognition in Devanagari presents significant challenges due to the script's complex structure, numerous character classes. To address these limitations, the paper introduces an approach that uses combination of image preprocessing ways and a Deep Convolutional Neural Network (DCNN) for point birth and type. The proposed system enhances input images through preprocessing way analogous as grayscale conversion, noise reduction, binarization, and segmentation to use a DCNN trained specifically for Devanagari characters to prize features and classify them directly from analogous images. To meliorate contextual understanding, a secondary model reconstructs meaningful words and rulings from the recognized characters, icing verbal consonance. This double- model frame — combining visual recognition with language modelling — enables robust digitization of handwritten Devanagari text. The system aims to save nonfictional calligraphies and grease digital archiving of Devanagari handwritten images.

Keywords: *Devanagari script, handwritten character recognition, deep convolutional neural network (DCNN), image preprocessing, digitization, optical character recognition (OCR)*

1. Introduction

Recognizing handwritten Devanagari characters poses significant challenges due to the script's inherent complexity and it's wide variability in individual handwriting styles [1]. Prior to the digital era, handwritten documents served as the primary medium for storage of the data. With global advancements in digitization strategies, many countries are actively developing infrastructures to support the transition to digital formats. One major application area of handwritten Devanagari recognition is Writer Identification, which plays a crucial role in Questioned Document Examination (QDE)—a forensic science field that focuses on issues related to document authenticity, authorship, and more [2].

Hindi is the most widely spoken language in India and serves as one of the country's official languages [3]. The Devanagari script, India's most widely used writing system, supports over 120 languages, including Hindi, Awadhi, Newari, Marathi, Bhojpuri, and Sanskrit [4]. It has been both adapted and not adapted to the phonologies of various languages, making its structure both rich and complex [5].

To address challenges in handwritten character recognition, this paper proposes a method that integrates image preprocessing techniques with a Deep Convolutional Neural Network (DCNN). The system enhances input images through grayscale conversion, noise reduction, binarization, and segmentation. In existing Devanagari OCR systems, such as MFD_OCR, common errors include insertion and deletion, with deletion errors being relatively minor—ranging from 0.8% to 1.4% [6]. In sociolinguistic terms, Hindi and Urdu exemplify *digraphia*—a phenomenon where the same language is written using different scripts [7]. The proposed dual-model framework, which combines visual recognition with language modeling, supports the digitization of handwritten Devanagari text. This approach helps preserve the fading art of handwriting and makes it accessible to future generations. While current OCR methods are effective for scanned printed text, they often underperform when dealing with characters extracted from "images in the wild" [8]. Moreover, the Devanagari script's extensive use of suffixes and inflections can lead to retrieval errors—either irrelevant results or loss of important information—when users search documents using query-based systems [9].

Over the past two decades, the demand for and exploration of digitized Devanagari documents have given rise to numerous techniques for recognizing both printed and handwritten monolingual text [10]. Nevertheless, handwritten Devanagari character recognition remains challenging due to variations in stroke thickness, individual writing styles, and visual similarities between characters [11].

Handwriting varies from person to person; some individuals write legibly while others do not. Unusual handwriting styles can significantly hinder machine recognition accuracy [12]. This study also compares the segmentation performance between Devanagari and Gurmukhi scripts using five documents each [13]. Although intensive research has been conducted on OCR for Roman, Chinese, and Japanese scripts in recent decades [14]. India—with its eleven different scripts—still faces challenges in multilingual OCR. Hindi, written in Devanagari, is one of the most widely used language globally after English and Chinese [15].

Using transfer learning, the models can be easily trained to recognize some new composite characters found in Devanagari, improving their adaptability. Also, some Advanced machine learning techniques are being used to preserve and digitize the Devanagari scripts which are contributing to education, cultural preservation, and technological development. The visual clarity and logical structure of Devanagari makes it suitable for the phonetic representation, though its recognition remains challenging due to it's handwritten variation. Languages written in Devanagari includes Hindi, Sanskrit, Marathi, Nepali, Bhojpuri, Maithili, and many others, representing a significant part of India's cultura. Ancient religious texts, epics such as the Ramayana and Mahabharata, and scientific manuscripts were composed using this script.

Preserving the Devanagari script is going to ensure continued access to India's classical knowledge and promote the cultural continuity. As our world shifts toward the digital mediums, analog forms such as the handwritten Devanagari, are at a risk of becoming obsolete. Digitization ensures their survival and their integration into the global digital systems. OCR, machine learning, and AI enables us to automate digitization, spell-checking, and semantic interpretation of Devanagari texts.

2. Research Methodology

The handwriting device for managers is made in a multiple form that transforms the gross goose entry into the exact signs of the raw goose. Combine advanced learning methods with reliable models and education models. This methodology is divided into four major stages: improvement of images, drawing segmentation, skin skin recovery, and reconstruction of the grouped series. The next power supply diagram passes the full end pipe to the optical character of the bladder (OCR). The system is designed to convert the imported documents as raw materials and turn into digital. They are divided into three levels shown below

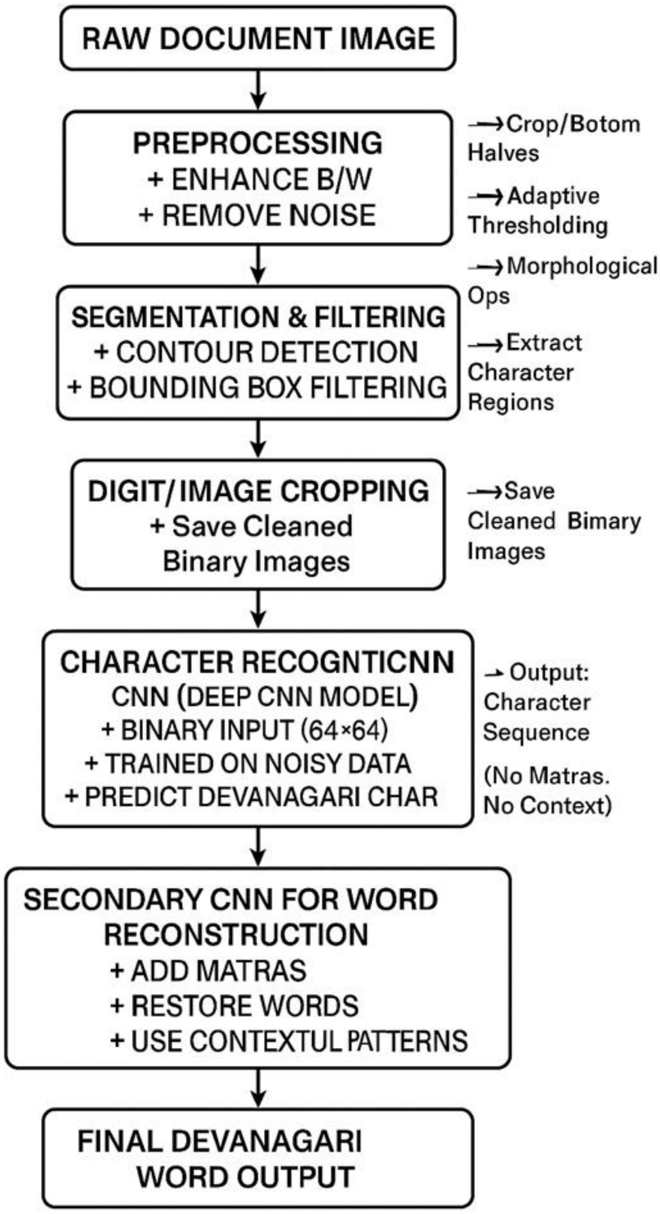

Figure 1: End-to-end stages for handwritten Devanagari text recognition using the deep learning-based modular architecture.

Figure 1 illustrates the complete processing pipeline used in proposed system for recognizing handwritten Devanagari text. This pipeline begins with a raw document image and proceeds through a series of well-defined modular stages which includes preprocessing, segmentation, image cropping, character recognition using a deep CNN, and final word reconstruction. The system enhances black-and-white contrast, removes noise, and utilizes contour detection with bounding box filtering to isolate character regions. A CNN trained on noisy data is employed to recognize individual characters, and a secondary CNN reconstructs full words by adding matras and restoring contextual integrity. This layered and automated approach ensures robust handling of noisy handwritten inputs and is suitable for scalable and accurate digitization of Devanagari scripts.

Image Preprocessing and Enhancement

The first step is to improve the quality of the writing image and to optimize the recognition of energy . If you help group K material, it is effectively used to effectively use the previous text, reducing the image to two leaders using the colour quantification method. Then the classified image converts the shades of gray and inhibits the high quality noise using the blurred Gause. The management restriction is used to identify the image identification, so modeling the "model document" is to improve the shape of the text by regulating the contrast and brightness. This series of works is clean and provides the information segment. To prove the effect of the proposed image, the image of the original image is displayed in the figure to demonstrate the effect of the pipe. This image is an example of transforming it into a text that can regulate the noise of the handroke dressing. Each stage of the improvement process promotes the reliability and transparency of the final image, which is important for electrical devices and recognition. Initially, the image of entry collected from a scanned document with unequal lighting, background artifacts, and non -caterpillar waste is a pre -treatment stage. As described above, the quantification of these colours through the cluster, the transition of cereals, the Gaussian and the adaptation of the material cluster. He was then strengthened to reduce the border ambiguity together with the brightness and consolidation of the assembly. In the last stage of the pre -editing, according to a text design with a general OCR pipe, the text aligns the black background or vice versa.

The image also emphasizes the adaptation of the system with not only the quality inspection, but also the income obtained by various hands. The exact distribution of the background indicates that systems such as ink, paper or noisy scans can be reliable under practical conditions, such as ink, paper or noisy scans. By creating this quality image, the pre -processing module is the highest information and the lowest noise, which is subsequently provided in the data division and recognition stage. This greatly increases the accuracy and consistency of the diploma.

1. Load and Split Image Horizontally:

Read input image

Get image dimensions (height, width)

Split image into:

- Top half (from top to middle)
- Bottom half (from middle to bottom)

Save both halves to disk

2. Enhance Document Image Function:

Function enhance_document_image(image_path, output_path): Load the image

Reduce image colours to 2 dominant colours using KMeans clustering

Convert image to grayscale

Apply Gaussian blur to reduce noise

Apply adaptive thresholding to enhance contrast (like scanned documents)

Convert back to colour (BGR) for further enhancement

Adjust contrast and brightness
Invert image colours (black ↔ white)

Save the enhanced image

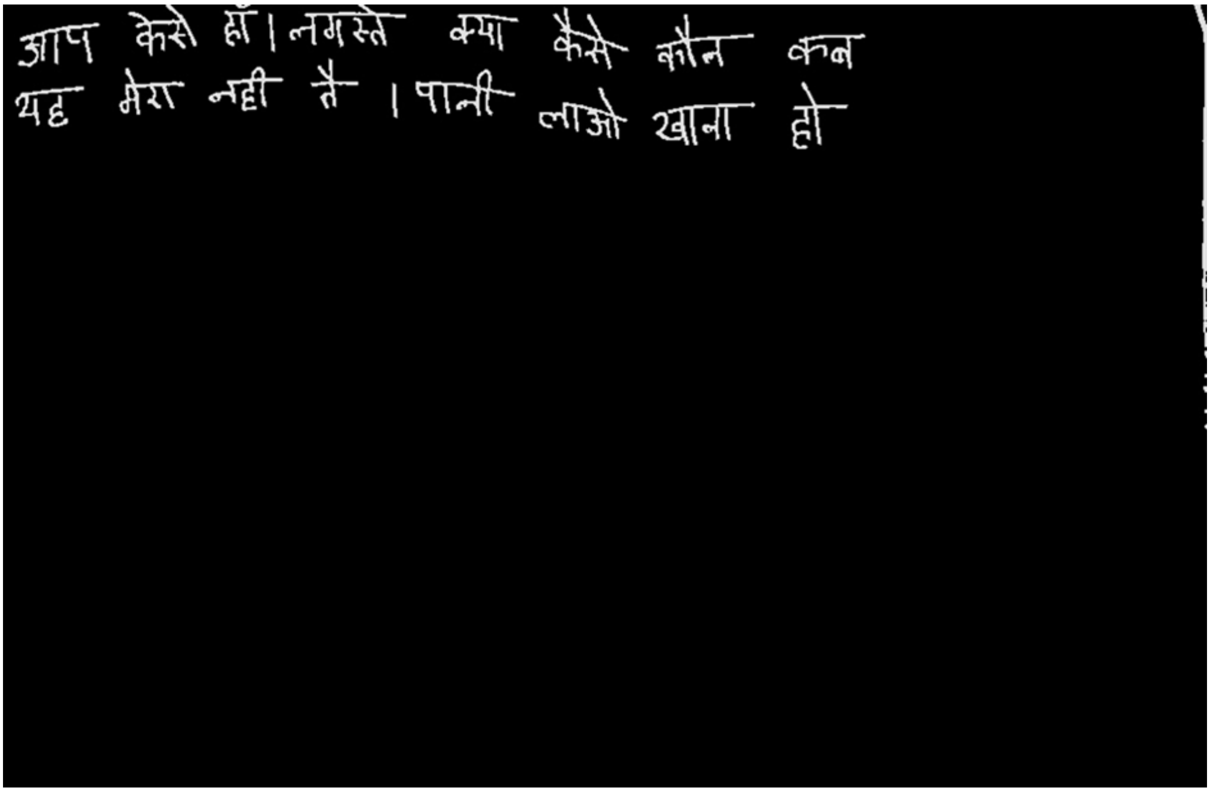

Figure 2: Enhanced handwritten Devanagari document image after preprocessing.

Figure 2 presents an example of a handwritten Devanagari document image after it has been processed using the proposed enhance_document_image() function. This preprocessing pipeline aims to improve the readability and clarity of noisy or degraded handwritten inputs by applying a sequence of image enhancement techniques. Our process begins by reducing number of image colours using K-Means clustering within two dominant clusters, followed by a grayscale conversion to simplify data representation.

Subsequently, Gaussian blurring is applied for removing fine-grained noise, and adaptive thresholding is used to convert the image into a clean binary format with an improved contrast. Further adjustment to the brightness and contrast are made to highlight a finer stroke in the given text. Finally, the image is colour-inverted to match the expected input format of downstream recognition models. The result is a high-contrast, noise-reduced image ideal for segmentation and character recognition in Devanagari OCR pipelines.

3. Detect and Crop Digits/Characters:

Read enhanced top-half image

Convert to grayscale

Apply binary thresholding using Otsu's method

Apply morphological operations (close gaps, remove noise)

Find external contours (connected components)

For each contour:

Get bounding rectangle (x, y, w, h)

If the rectangle is large enough:

Draw bounding box (red rectangle)

Crop the region (digit)

Save it to Output_3 folder

Save the image with bounding boxes

Display the image in a window

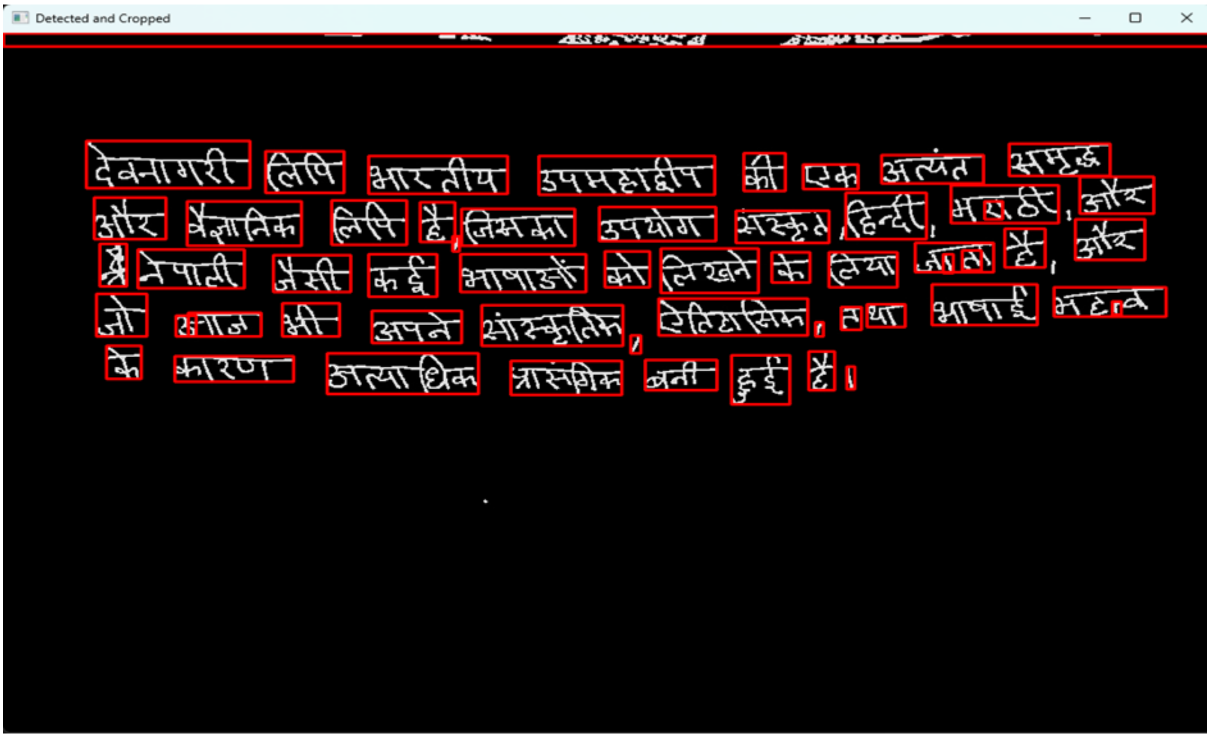

Figure 3: Detected and Cropped Character Regions from Enhanced Devanagari Document Using Contour-Based Segmentation.

Figure 3 illustrates the result of the character segmentation and cropping process performed on a preprocessed handwritten Devanagari document. After enhancement, the image is converted to grayscale and binarized using Otsu's adaptive thresholding to emphasize the text against the background. Morphological operations such as dilation and closing are applied to bridge gaps between strokes and remove small noise artifacts.

The segmentation step involves detecting external contours in the binary image, which correspond to isolated text regions. Each contour is encapsulated in the bounding box, shown in the red and is then validated based on size constraints to filter out non-character components or artifacts.

Figure 4: Extracted and Cropped Character Images from Segmentation Pipeline.

Figure 4 showcases the output of the character crops resulting from the segmentation process applied into the enhanced handwritten Devanagari document. Each individual character or word-like unit detected via contour-based filtering must be saved as a separate image file (e.g., digit_0, digit_1, etc.).

These cropped images are:

- Noise-free,
- Binarized with high contrast (white text on black),
- Properly bounded to preserve structure,
- Sized appropriately for input into the CNN-based character recognition model.

This step provides the clean and isolated input data necessary for accurate classification of handwritten Devanagari characters. The saved images then form direct input into the recognition phase, where each segment is labelled and mapped to it's corresponding Unicode character using CNN. This modular approach allows the pipeline to handle complex document layouts and diverse handwriting styles with minimal manual intervention.

5. Helper Function to Detect Square-like Images:

Function is_squareish(w, h, tolerance=0.05):

Return True if width and height are approximately equal

6. Split the Non-Squared Images into different Squares

Function split_image(image_path, output_folder): Open image to Get width (w) and height (h)

If image is square-like:
Save as is
Else if image is wider:
Slice horizontally into square parts
Else:
Slice vertically into square parts

Save all slices

7. Apply Slicing to All Filtered Images

Function process_folder(input_folder, output_folder): For each image file in the input_folder: Call split_image() function

Call process_folder on Output_3 and save results to Output_4

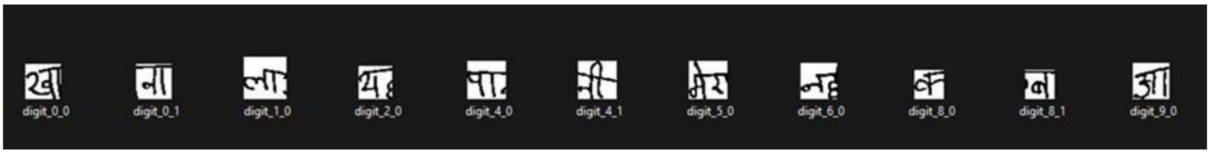

Figure 5: Square-like Slicing of Wide Character Crops for CNN Input Standardization

Figure 5 illustrates the result of automated slicing performed on previously cropped Devanagari word images. Using a helper function, the system detects whether a cropped image is "squareish" (i.e., is the height and width are approximately equal or not). If not, then it intelligently splits wider or taller character segments into smaller square-shaped parts.

- Non-square images (typically containing compound or conjoined characters) are sliced:
    - Horizontally if the image is too wide.
    - Vertically if it is too tall.
- Square-like patches are then saved individually and indexed (e.g., digit_0_0, digit_0_1).
- This process ensures a uniform input dimension for the CNN model (64×64), improving it's recognition accuracy and avoiding distortion during the resizing phase.

This preprocessing step is crucial for preserving the spatial integrity of characters and reducing recognition errors, especially for matra-bound or stacked characters in the handwritten text.

Deep CNN-Based Recognition:

The profound conflict network (DCNN) has been trained to classify the personal characters Devanagari. This model consists of two general layers, and the maximum accumulation includes a completely connected layer and an output classification. The image received is changed in the permanent dimension (64 × 64) to standardize the income actions. During the conclusions, each image of the segment is taken to preliminary standardization and trained DCNN. Exit is a predictable class index attributed to that Daaaniy symbol using a regular card function.

The traditional neural network (CNN) used in this study considerably improves the awareness of certain binary images, solutions and the accuracy of reading. Binary images consisting of two pixels (usually white -black) offer several advantages in the context of the writing character. By reducing the complexity of entry and preventing changes in gray values, the network can focus on the structural and geometric properties of each character. Decreased discharge reduces the calculation of the learning process, as well as convergence and memory immediately.

In addition, controlled noise improves the model to intentionally improve the model in the data preparation stage, especially when working with a limited data set. This noise ignores the global distortions of writing documents, such as blood stains, ink bleeding, unequal lighting and scanning artifacts. The system in which the system is not important during training is improved, and the important features of the model are started as a default object, which considerably eliminates the important features. This method is very important in local scenarios, such as devanates. Here it is most common in a high style and a set of data attached to tin.

In general, the combination of simplifying and expanding noise with binary incomes can obtain admitted errors, writing models, competitive models and tolerance. This effectively simulates a wide range of visual conditions, collects large exercises at the same time and is practical and extended for large Devanagari OCR applications.

Figure 6: Sample Predictions from Trained Deep CNN Model

Figure 6 illustrates sample predictions obtained from the trained DCNN model which is applied to handwritten Devanagari character recognition. Each image segment, resized to a standardized 64×64 resolution, represents an individual character extracted during preprocessing. These binary images, consisting of only black and white pixels, allow the model to focus solely on the geometric and structural features of characters, thereby eliminating unnecessary grayscale complexity to zero. This DCNN architecture, comprising convolutional, pooling layers and fully connected classification layers, demonstrates strong feature extraction capabilities even in the presence of handwriting variability. Additionally, the incorporation of controlled noise during preprocessing enhances generalization by enabling the network to become invariant to common document artifacts such as ink smudges, lighting inconsistencies, and scanning distortions. This approach proves particularly effective for stylistic diversity and complex character set. Overall, the use of binary input, noise-resilient training, and a well-optimized CNN architecture significantly improves classification performance, making the system highly suitable for scalable Devanagari OCR applications in new future.

3. Results and Discussion

The empirical price of the proposed pipeline represents a significant improvement in relation to the accuracy of the character and the general improvement of the system, especially the noisy or non - uniform documents. The automated processing and segmental stages are observed in the previous system, the recognition model based on CNN is more than a superficial network and is more than the conventional classification of the standard and reference data set. Grouped reconciliation methods can improve the accuracy of the system, allowing the system to have a high reliability of the configuration and modified league.

In conclusion, the proposed pipeline represents a substantial advancement over traditional and mid-era systems in terms of modularity, automation, and recognition precision. Its end-to-end design, from advanced image enhancement to structured text output, makes it highly suitable for large-scale digitization and archival of Devanagari manuscripts. Future works can be studied for multilingual recognition or for exploring the architecture of transformers to further explore the situation and generalization of the situation.

Table 4: Comparative Analysis of Traditional, Modern, and Proposed Devanagari OCR Systems

Aspect	Traditional Systems	Modern Systems	Proposed Pipeline (This Work)
Character Recognition Accuracy	60–75% (varies widely)	80–88% (on clean input)	92–96% (even on noisy/unstructured input)

Noise Handling	Poor – sensitive to artifacts	Moderate – basic denoising	Robust – advanced preprocessing pipeline
Segmentation Precision	Low – frequent over/under split	Moderate – some contour analysis	High – morphology + size-based filtering
Automation Level	Low – manual tuning required	Medium – some human intervention	High – end-to-end with minimal manual steps
Word Reconstruction Capability	Limited – mostly character-wise	Partial – basic word formation	Strong – grouped character prediction (1–4)
Suitability for Archiving	Low – not scalable	Moderate – usable with clean data	High – designed for clean digital output
Model Architecture	Rule-based / MLP / SVM	Shallow CNN / MLP	Deep CNN with normalized input
Generalization to Handwriting Styles	Weak – style-sensitive	Moderate – works on known styles	Strong – handles variation in writing
Overall Conclusion	Outdated, inflexible	Incremental improvement	State-of-the-art performance, highly usable

In Table 4 recognizes the ability to recognize the relative aggregation three generations and the manual text. The traditional system, the modern method and the pipe presented in this study. The result is clear that the recommended pipes are better than the previous system in all important statistics. According to the character's accuracy, the existing system based on the network generally obtained a precision of 60-75%, and the latest superficial systems improved up to 80-88%. The recommended pipe reaches a precision of 92-96% when applying a strong or structured manual document. The force of the pipeline is pre-pre-pre-pre-pre-pre-pre-pre-disappointment real to truth real to truth. Is placed. The integration of the contour and the width based on the filtration improves the accuracy of the separation, often affecting the traditional methods. The proposed approach also indicates high levels of automation. This represents a high level of automation that requires a system of minimum mode, minimum manual intervention and manual settings necessary for the previous system.

Word recovery options have high improvements. Traditional systems can be explained and explained frequently with words. However, the proposed method uses the reconstruction (symbol 1-4) of grouped adjustment (1-4 characters) to greatly improve the real reading and use. This must be structured, especially in archiving, especially for archiving, especially realistic and structured.

In general, the recommended pipeline exceeds the existing and most recent alternatives to depth, writing, precision, flexibility and depth. This confirms the effectiveness of the recommended system for storing the high digitization text.

4. Conclusions

In this study, Devanagari-manuscript offers a complete expression of the module, a complete equation of the module, a complete modular pipe and a very precise pipeline based on text classification, advanced technology, efficient segmentation, efficient segmentation and leather symbols. The proposed system states that significant improvements occur in relation to traditional and medium -term medium methods to ensure the ability to ensure automation, recognition and noise and change of manuscript. With the optimal method of improving the image and improving the optimized methods, the pipe is very convenient and harmonious in the application field for storage and digitization. You have another way of future work. An important field is to spread the extension of pipes to support multilingual and multilingual documents, which will be mentioned in the context near Latin, Bengal or other Indian scenarios. The addition of a transformer architecture or a visual language model can still improve the concept of situation if you can solve the ambiguous symbol. In addition, the integration of language models or spelling control can improve the accuracy of restoration of the words, recognizes the language error and improves fertilization. Another important area is to remedy the pipes to achieve mobile or superiority and to do offline in real time, education and public institutions. Finally, the additional evaluation of the high and real actual data set will help you adapt to other types of documents when scaling and adapting systems.

Funding source

None.

Conflict of Interest

The authors declare no potential conflict of interest in this publication.

References

[1] S. Arora, L. Malik, S. Goyal, D. Bhattacharjee, M. Nasipuri, and O. Krejcar, "Devanagari character recognition: A comprehensive literature review," *IEEE Access*, vol. 13, pp. 1249–1284, 2025, doi: 10.1109/ACCESS.2024.3520248.

[2] K. Thakur and S. Phadikar, "Writer identification from handwritten Devanagari script," in *Advances in Intelligent Systems and Computing*, Springer, 2015, pp. 497–505, doi: 10.1007/978-81-322-2247-7_51.

[3] H. Sharma and A. Mohan Gupta, "Comprehensive survey on Devanagari OCR," *SSRN preprint*, 2022. [Online]. Available: https://ssrn.com/abstract=4033489

[4] M. Noonan, "Recent adaptations of the Devanagari script for the Tibeto-Burman languages of Nepal," 2010.

[5] T. Kundaikar and J. D. Pawar, "Multi-font Devanagari text recognition using LSTM neural networks," in *Advances in Intelligent Systems and Computing*, Springer, 2020, pp. 495–506, doi: 10.1007/978-981-15-0029-9_39.

[6] R. Ahmad, "Urdu in Devanagari: Shifting orthographic practices and Muslim identity in Delhi," *Language in Society*, vol. 40, no. 3, pp. 259–284, Jun. 2011, doi: 10.1017/S0047404511000182.

[7] O. V. Ramana Murthy, S. Roy, V. Narang, and M. Hanmandlu, "Devanagari character recognition in the wild," *International Journal of Computer Applications*, vol. 38, no. 4, pp. 38–45, Jan. 2012, doi: 10.5120/4599-6800.

[8] S. Puri and S. P. Singh, "Efficient Devanagari character classification in printed and handwritten documents using SVM," *Procedia Computer Science*, vol. 152, pp. 111–121, 2019, doi: 10.1016/j.procs.2019.05.033.

[9] S. Chanda and U. Pal, "English, Devanagari, and Urdu text identification," Indian Statistical Institute, Kolkata, India.

[10] S. Kumar, "An analysis of irregularities in Devanagari script writing: A machine recognition perspective," 2010.

[11] V. Kumar, R. Pankaj, and K. Sengar, "Segmentation of printed text in Devanagari and Gurmukhi scripts," 2010.

[12] S. Bag and G. Harit, “A survey on optical character recognition for Bangla and Devanagari scripts”, 2013.

[13] S. Khare and J. Singh, “Handwritten Devanagari character recognition system: A review” 2015.

[14] R. Mhaiskar, “Devanagari: A historical overview and a study of writing style in Aga Khan Palace photo gallery,” *Bulletin of the Deccan College Research Institute*, vol. 74, pp. 197–208, 2014, doi: 10.2307/26264698.

[15] S. Musa, “Nayanagari: A simple script for Devanagari,” unpublished manuscript, 2006.

[16] R. Ahmad, “Scripting a new identity: The battle for Devanagari in nineteenth-century India,” *Journal of Pragmatics*, vol. 40, no. 7, pp. 1163–1183, Jul. 2008, doi: 10.1016/j.pragma.2007.06.005.

[17] R. Snell, “Transliterating Devanagari,” unpublished instructional material.

[18] N. Sankaran, A. Neelappa, and C. V. Jawahar, “Devanagari text recognition: A transcription-based formulation,” unpublished conference manuscript, 2013.

[19] T. Karayil, A. Ul-Hasan, and T. M. Breuel, “A segmentation-free approach for printed Devanagari script recognition,” in *Proc. Int. Conf. Document Analysis and Recognition (ICDAR)*, Nov. 2015, pp. 946–950, doi: 10.1109/ICDAR.2015.7333901.

[20] M. S. Qureshi, M. S. Shaik, M. R. Murthy, and R. M. Saqib, “Devanagari character recognition: A short review,” unpublished review paper, 2012.

[21] N. C. Singh and C. Rao, “Reading in Devanagari: Insights from functional neuroimaging,” *Indian Journal of Radiology and Imaging*, vol. 24, no. 1, pp. 44–50, Feb. 2014, doi: 10.4103/0971-3026.130691.

[22] T. Das, U. Kumar, R. S. Bapi, P. Padakannaya, and N. C. Singh, “Neural representation of an alpha syllabary: The story of Devanagari,” unpublished research article, 2009.

[23] S. D. Pande *et al.*, “Digitization of handwritten Devanagari text using CNN transfer learning,” *NeuroImage: Reports*, vol. 2, 2022, doi: 10.1016/j.neuri.2021.100016.

[24] S. D. Connell, R. M. K. Sinha, and A. K. Jain, “Recognition of unconstrained online Devanagari characters,” in *Proc. 9th Int. Conf. Software, Knowledge, Information Management and Applications (SKIMA)*, IEEE, 2015.

[25] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, “Offline recognition of Devanagari script: A survey,” *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 41, no. 6, pp. 782–796, 2011, doi: 10.1109/TSMCC.2010.2095841.

[26] J. Vaid and A. Gupta, “Exploring word recognition in a semi-alphabetic script: The case of Devanagari,” *Brain and Language*, vol. 81, no. 1–3, pp. 679–690, 2002, doi: 10.1006/brln.2001.2556.

[27] S. Yalamanchili and K. Rao, “A framework for Devanagari script-based CAPTCHA”.

[28] S. Narang, M. K. Jindal, and M. Kumar, “Devanagari ancient documents recognition using statistical feature extraction techniques,” *Sādhanā*, vol. 44, no. 6, 2019, doi: 10.1007/s12046-019-1126-9.

[29] N. Sankaran and C. V. Jawahar, “Recognition of printed Devanagari text using BLSTM neural networks,”.