IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 November 2024, accepted 13 December 2024, date of publication 19 December 2024,
date of current version 2 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3520248

RESEARCH ARTICLE

Devanagari Character Recognition: A Comprehensive Literature Review

SANDHYA ARORA[1], LATESH MALIK[2], SONAKSHI GOYAL[1], DEBOTOSH BHATTACHARJEE[3], (Senior Member, IEEE), MITA NASIPURI[3], (Life Senior Member, IEEE), AND ONDREJ KREJCAR[4]

[1]Cummins College of Engineering for Women, Pune 411052, India
[2]Government College of Engineering, Nagpur 441108, India
[3]Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, India
[4]Research Center, Skoda Auto University, 293 01 Mlada Boleslav, Czech Republic

Corresponding author: Latesh Malik (latesh.gagan@gmail.com)

This work was supported in part by the Long-Term Conceptual Development of Research Organization (2024) at Skoda Auto University, Czech Republic.

ABSTRACT The Devanagari script originated from the ancient Brahmi script and is a widely used Indic script for writing different languages, like Sanskrit, Hindi, Marathi, Nepali, and Konkani. Recognizing handwritten Devanagari characters poses significant challenges due to their complexity and handwriting variability. This literature review examines the evolution of handwritten Devanagari character recognition (HDCR), exploring early template matching and feature extraction methods that struggled with the script's intricacy. Advances introduced structural and statistical techniques, improving accuracy by analyzing geometric properties and patterns. The advent of machine learning, particularly deep learning, revolutionized HDCR with convolutional neural networks (CNNs) and recurrent neural networks (RNNs), significantly enhancing performance. Hybrid approaches that combine multiple techniques have shown promising results, balancing accuracy and computational complexity. Challenges remain, including handwriting variability, noise, and the need for real-time performance. The lack of large, diverse datasets for training and evaluation is a significant hurdle. This review highlights efforts to create annotated datasets and benchmarks, providing a comprehensive overview of HDCR methodologies, strengths, limitations, and future research directions. These insights aim to advance HDCR, contributing to more accurate and efficient recognition systems and enhancing digital text processing for linguistic, educational, and archival purposes.

INDEX TERMS OCR, handwritten Devanagari character recognition, machine learning, deep learning.

I. INTRODUCTION

Handwritten documents were utilized for data storage before the introduction of computing systems. A handwritten paper contains massive data for land records, parental information, birth information, holy books, study material, and many other things. Most ancient records in emerging countries are still in the form of printed or handwritten papers.

Character recognition was made possible by Carey's invention of the retina scanner, an image transmission system based on a mosaic of photocells, in 1870 [1]. Nipkow devised the sequential scanner in 1890, a critical development for contemporary television and reading machines. However, character recognition was once thought to be a tool for the visually impaired, and the first effective attempts were conducted in 1900 by the Russian scientist Tyurin.

Commercial OCR systems are classified into four generations based on adaptability, resilience, and efficiency. The limited letter shapes that OCRs read characterize first-generation systems. Such machines first appeared in the early 1960sThe IBM 1418, designed to read the exclusive IBM typeface 407, was the first widely available OCR of its generation [2]. Logical template matching was the recognition strategy, using the positional relationship.

The associate editor coordinating the review of this manuscript and approving it for publication was Yu-Da Lin.

The capacity to distinguish between handwritten and conventional machine-printed characters sets apart the following generation. The scope was first restricted to numbers. The earliest of these machines appeared in the late 1960s and early 1970s. This generation's earliest and most well-known OCR system was IBM 1287, which was on show during the 1965 New York World's Fair [2]. The system's hardware combined digital and analog technologies in a hybrid design. Toshiba created the first automated letter-sorting system for postal code numbers during this period. The procedures were developed using the structural analysis methodology.

Low print quality character OCR and hand-printed characters for a broad category of characters set the third generation apart. From 1975 to 1985, commercial OCR systems with comparable features first became available [2], [3], [4].

OCR of complex documents containing text, images, mathematical and tabular symbols, unrestricted handwritten characters, color documents, loud, low-quality documents from copier and fax machines, and other sources set apart the fourth generation. Some complex document work produced positive results. Even though there are many works on unconstrained handwritten characters in the literature, research is still ongoing. Researchers are still working on noisy documents [5], [6] and color papers.

Postal address readers, among other commercial items, are available on the market. Around 60% of hand-printed material in the United States is sorted automatically [7]. There is also a reading aid for visually impaired people. Xerox-Kurzweil has marketed an integrated OCR with a speech output system in English for visually impaired people [8].

More sophisticated optical readers for Roman, Chinese, Japanese, and Arabic text are now available [9], [10], [11], [12], [13], [14], [15]. These readers can read typewritten, typeset, or printed texts from dot-matrix, line, and laser printers. They can recognize characters in various fonts, sizes, and forms, including combined text and pictures. Columnar scanning is now possible because of the arrival of narrow-range scanners measuring 3 to 6 inches wide. Using these scanners, an optical reader can recognize several columns or portions of a page or mailing list. Some have software for spell-checking and identifying questionable characters or words [16].

The Indian constitution lists the eighteen official languages of India as Assamese, Bangla, English, Gujarati, Hindi, Konkani, Kannada, Kashmiri, Malayalam, Marathi, Nepali, Oriya, Panjabi, Rajasthani, Sanskrit, Tamil, Telugu, and Urdu. Recently, the list was expanded to include a few more languages. The two most widely spoken languages in India are Hindi and Bangla, which rank fourth and fifth in the world, respectively. Many modifications separated Indian scripts from the ancient Brahmi script [17]. Two or more of these languages may share the same script. Devanagari, for example, is used to write Hindi, Marathi, Rajasthani, Sanskrit, and Nepali, while Assamese and Bangla (Bengali) languages use Bangla script.

Marathi	देवनागरी ही भारतीय लिपी आहे
English	Devnagari is an Indian script
Kannada	ದೇವನಾಗರಿ ಭಾರತೀಯ ಲಿಪಿ
Bangla	দেবনাগরী একটি ভারতীয় লিপি
Hindi	देवनागरी एक भारतीय लिपि है
Telgu	దేవనాగరి ఒక భారతీయ లిపి

FIGURE 1. Examples of a few indian scripts.

A. DEVANAGARI CHARACTER SET

As illustrated in Figure 2, the Devanagari script alphabet system consists of 49 fundamental letters, comprising 13 vowels and 36 consonants. Compound characters can be made by combining two or more basic characters, as well as vowel and consonant characters, which make up basic characters. A compound character's shape is more complex than its basic characters. There are several ways to combine basic characters to form a basic compound character. One method is to cut off the character's vertical line and join it with the other character from the left side. Another method is to unite the characters side by side or one above the other. The compound's e.g., is displayed in Figure 4.

It may take on a different shape depending on whether a vowel comes after a consonant to the left, right, top, or bottom. We refer to them as modified characters. The term ''shirorekha'' or header line refers to the horizontal line at the top of several Devanagari characters. A broad head-line is formed in linguistics when two or more characters are put side by side to make a word. Character separation from words gets tricky as a result. Three zones are used to write Devanagari characters: ''upper matras or modifiers'' are written in the top zone, which is located above the horizontal bar (shirorekha or header line); the word itself is written in the middle zone, and ''lower matras or modifiers'' are written in the bottom zone. The existence of compound characters and modifiers makes identifying Devanagari characters harder and trickier. The modifier and the three Devanagari character zones are displayed in Figure 3. Due to ligatures, context-dependent forms, and stylistic variations, unique challenges arise for handwritten Devanagari character recognition.

It is possible to recognize handwritten Devanagari script characters online or offline [17]. Techniques for online recognition and offline recognition can be generated by online or offline data collection methods, respectively. While online identification techniques use digital devices like PCs, tablets, and electronic pads, offline recognition techniques use scanned picture files. The benefit of online systems is that critical data, such as pressure points, writing trajectories, and stroke directions, may be swiftly and precisely recorded. However, calculating strokes in offline systems is challenging since they rely on scanned input images [18]. Because online characters are based on temporal aspects, including form, stroke count, writing direction, and distance, identifying them

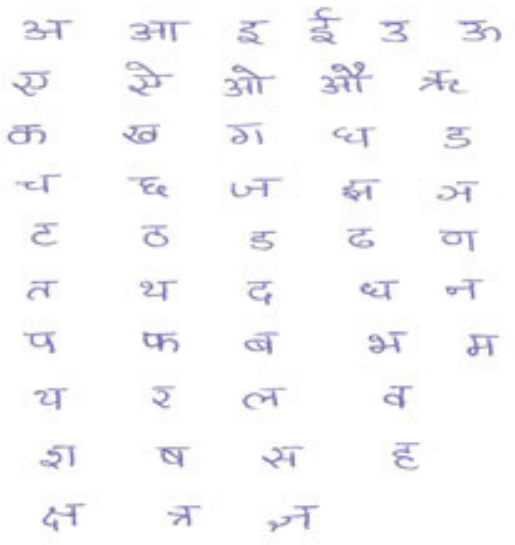


FIGURE 2. Basic devanagari characters.

FIGURE 3. Three zones of devanagari characters.

FIGURE 4. Examples of compound characters of devanagari script.

is relatively easy. Implementing offline character recognition is complex due to variations in writers and fonts.

Compared to online systems, offline systems have lower accuracy. But today, they are advanced enough to play a significant economic role in particular fields, like reading bank cheques' courteous amounts and deciphering handwritten postal addresses on envelopes. Because online systems are so successful, it is appealing to think about creating offline systems that use online recognition algorithms after estimating the writing's trajectory using offline data [19]. However, there aren't many of these feature extraction methods because it's hard to recreate the temporal data [20], [21], and [22]. For online character recognition, a plethora of techniques and strategies have been put out, evaluated, and compiled in a few comprehensive survey studies [23], [24], [25], [26].

This survey discusses the most implemented techniques for Hindi handwritten characters starting from the early years and highlights where Devanagari recognition techniques stand.

B. CHALLENGES

The challenge of recognizing handwritten Devanagari characters can be difficult due to various factors, including the intricate structure of the characters with their modifiers and the existence of compound characters. In the Devanagari script, writing is done from left to right. The Devanagari script lacks the idea of upper/lower case. A vowel after a consonant in Devanagari script has a different form. The modified shape of a vowel is positioned at the bottom, left, or right of the consonant, depending on the vowel. We refer to these altered forms as modified characters. Sometimes, a consonant or the vowel that comes after a consonant takes on a compound orthographic shape that we refer to as a compound character.

Here are some challenges associated with Devanagari character recognition:

1) COMPLEX SCRIPT STRUCTURE

Devanagari script has a complex structure with characters and various combinations of consonants and vowels. Ligatures and conjuncts are common, where two or more characters combine to form a single character. Hindi characters have complex shapes, as shown in Fig. 4. Hindi characters can be bifurcated into three zones: the upper zone, which is above the horizontal bar, consists of 'upper matras or modifiers,' the middle zone contains the entire word, and the lower zone comprises 'lower matras or modifiers' as illustrated in Fig 3. Most of the literature describes character segmentation in steps. It provides an overview of the basic steps required for character recognition. It also highlights the frequently used techniques of each stage [27].

Segmentation is completed by separation of the individual characters of a picture. Typically, a document is processed in a very hierarchical manner. Initially, level lines are divided by the victimization row bar chart. From every row, words are extracted from victimization column bar charts, and at last, characters are extracted from words [28]. The challenge of a segmentation technique lies in deciding the best segmentation point for line, word, and character isolation.

FIGURE 5. Overlapping characters [30].

FIGURE 6. Touching characters [30].

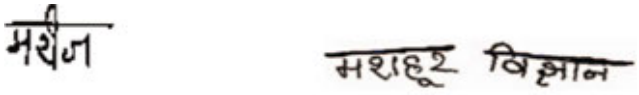


FIGURE 7. Regions overlapping [30].

In the handwritten script, many features are available for segmentation provided by a skeleton and the intersection point. Incorrect segmentation can lead to incorrect recognition. Segmentation of handwritten text is a difficult task owing to a variety of writing styles [29].

2) OVERLAPPING AND TOUCHING CHARACTERS

Because of incorrect handwriting, it is tough to recognize handwritten characters in Devanagari script. Sometimes, there is no gap between the characters when they are written over or close to one another. This makes it challenging to separate and distinguish each distinct character from the overlapped ones. Consonants, vowels, auxiliary signs, half characters, etc., are all part of the Devanagari character set. Because there isn't any vertical white space between the two characters, it becomes more complex and computationally demanding to segment these characters precisely. Thus, it is a very tedious job to segment and identify such overlapping characters in the Devanagari text, and next, in the classification process, it gets rejected as no such character is present in the database [30]. These are segmented as a single character [30].

3) REGION OVERLAPPING

These regions often overlap when writing the documents, as shown in Figure 7. This overlapping results in problems for the character recognition process. Also, the lower region of the upper string gets inserted into the upper region of the lower string, making it difficult to segment.

4) LARGE CHARACTER SET

Devanagari script has a more extensive character set than Latin scripts, making recognition more challenging. Different variations of characters may exist based on their context within a word.

5) INHERENT AMBIGUITY

Some characters in Devanagari look similar and can be visually ambiguous. Discrimination between characters with similar shapes becomes a challenge, especially in low-resolution or noisy images. Variable Writing Styles and Handwriting styles lead to diverse representations of the same character. Variability in font styles and distortions further complicates recognition.

6) LIMITED ANNOTATED DATASETS

The availability of large, annotated datasets for Devanagari character recognition is limited, making it challenging to train accurate models.

7) NOISE AND DISTORTIONS

Images of Devanagari text may have noise, distortions, or variations in lighting conditions, impacting the accuracy of recognition systems.

8) FONT, SIZE, AND WRITING VARIATIONS

Devanagari characters can be written in different fonts and sizes, making it challenging to develop models that generalize well across diverse styles. Most problems arise due to other styles of writing of all the persons as all have their style of writing, so some write characters or words of different font sizes and have their style of writing some words which become difficult for a machine to recognize some of those characters [30]

9) MULTILINGUAL CHALLENGES

Devanagari script is used for multiple languages with nuances and variations, challenging a universal recognition system.

Addressing these challenges often requires a combination of advanced machine learning techniques, large and diverse datasets, and domain-specific knowledge to develop robust Devanagari character recognition systems. Researchers and developers continue to improve these systems through advancements in machine, deep learning, and computer vision.

C. MOTIVATION

A few studies have been conducted recently on utilizing traditional methods, machine learning, and Deep Learning techniques for handwritten Devanagari character recognition [31], [32], [33], [34], [35]. This research briefly concentrates on several models applied to Devanagari HCR. This research aims to shed light on the HDCR process, including the methods employed and documented at each stage. Knowledge of the reported strategies would facilitate handling novel problems and proposed improved algorithms. We carefully studied each work to comprehend the methodology, benefits, and drawbacks and ascertain how and why specific approaches are practical in various contexts. A more profound comprehension of the benefits and disadvantages of these

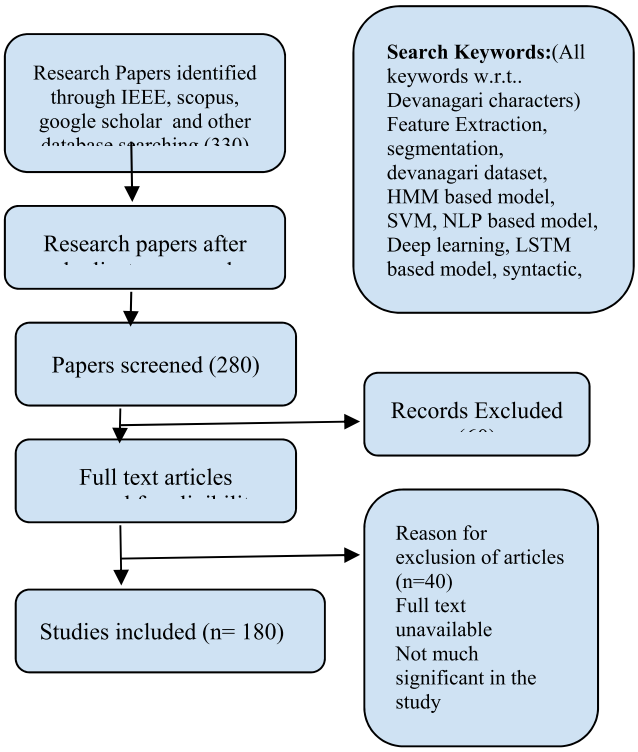


FIGURE 8. PRISMA flow chart demonstrating the literature search process.

strategies might lead to the development of new designs that maximize the advantages while avoiding their drawbacks.

D. REVIEW PROCESS

A systematic literature search was undertaken utilizing IEEE, Scopus, Google Scholar, and other databases to discover relevant articles for this complete review. The research papers were chosen using the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) method, which includes a flow diagram and a checklist. Fig. 8 shows the flow diagram and the number of research papers included or excluded at each stage. After following some observations, a title/abstract/keyword search was conducted in IEEE, Scopus, and Google Scholar using a search query, some of which are shown in Fig. 8. We targeted keywords related to Devanagari script recognition. Our inclusion criteria focused on papers published from 1976 to 2023 that presented novel algorithms or preprocessing, segmentation, and feature extraction techniques. To begin, 330 research papers are collected by searching databases. These documents are first screened, and 50 duplicates are eliminated. The remaining 280 records were chosen for abstract evaluation. After carefully reviewing the 280 papers and determining their eligibility, 60 papers were considered irrelevant to this meta-analysis and excluded. This rigorous approach allowed us to provide a detailed synthesis of current research and highlight directions for future investigations.

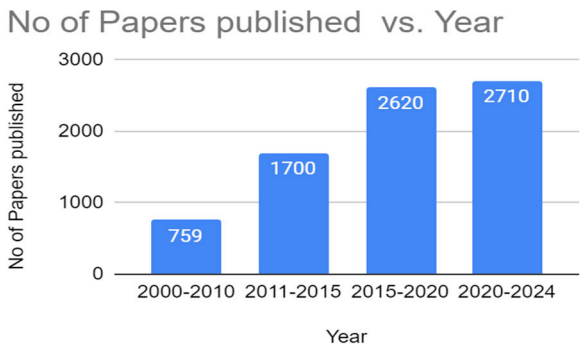


FIGURE 9. Publication Trends of Research Papers in the domain of Devanagari character recognition from 2000-2024.

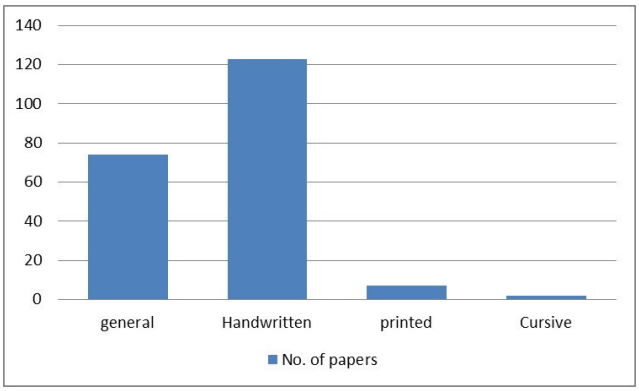


FIGURE 10. Distribution of referred research papers by type of text document.

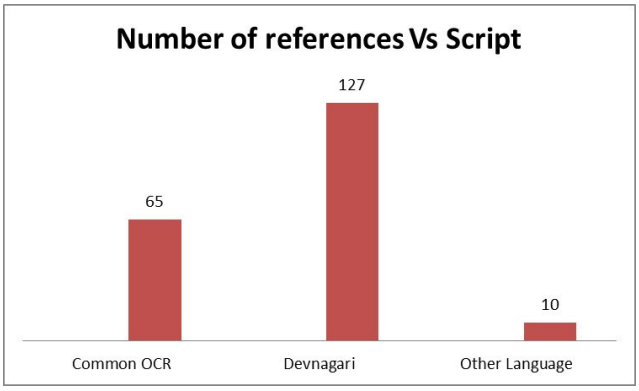
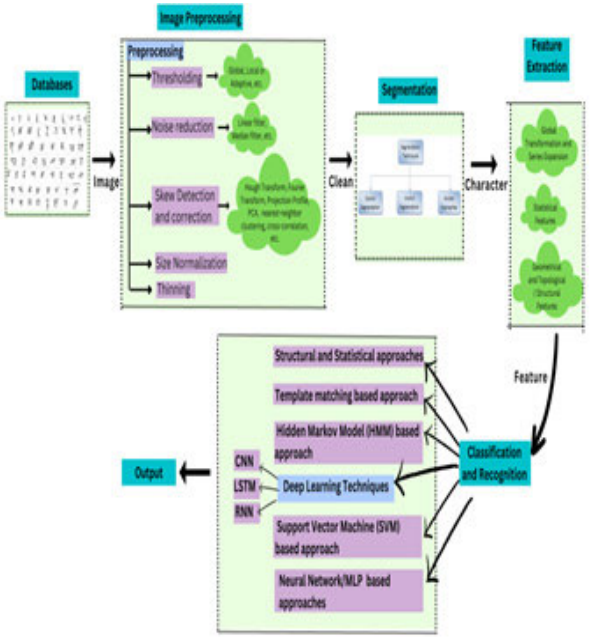


FIGURE 11. Script-wise distribution of referred papers.

The eligible 220 papers are analyzed, and 40 papers are excluded due to unavailability of the complete text, insufficient length, or lack of significant relevance to the study. These criteria ensured a focused and meaningful selection of research. Finally, only 180 research papers remain referenced in this article. Fig. 9 illustrates the distribution of research papers published across four time periods: 2000-2010, 2011-2015, 2015-2020, and 2020-2024 in the domain of Devanagari character recognition. Fig. 10 shows the



(a)

Elements/ Techniques	Characteristics/ Functionality
Preprocessing	
Thresholding	Converts the image to a binary format to separate foreground and background, enhancing contrast and simplifying further processing.
Thinning	Reduces line width to a single-pixel width while preserving the structure, making it easier to extract relevant features.
Noise Reduction	Removes unwanted artifacts like specks or irregularities, enhancing image clarity for more accurate recognition.
Size Normalization	Rescales the image to a standard size, ensuring uniform input dimensions for consistency in further processing.
Skew Correction	Aligns tilted or misaligned characters to improve accuracy in feature extraction and classification.
Segmentation	
Explicit Segmentation	Dissection to find boundaries of upper & lower modifiers, header lines and characters.
Implicit Segmentation	Recognition based matching with given characters
Holistic Segmentation	Segmentation free process to recognize the whole word.
Feature Extraction	
Global Transformation and Series Expansion	Applies mathematical transformations such as Fourier, Discrete Cosine Transform (DCT), Wavelet or expansions to represent characters in a reduced form that preserves essential shape information. This helps in capturing global patterns across characters.
Geometrical and Topological / Structural Features	Extracts features based on geometric and structural properties, such as strokes, curves, junctions, endpoints, chaincodes and loops. Effective for characters with distinct structures and patterns, helping recognize well-defined shapes.
Statistical Features	Uses structural and syntax based features like strokes, pixel distribution, gradients, moments curves, junctions and rule-based methods to recognize character patterns. Effective for well-structured, clearly segmented characters but less adaptable to diverse handwriting styles
Ensembled Feature Extraction Techniques	Combines multiple feature extraction methods (e.g., CNN, deep learning models with statistical or structural features) to capture a broader range of patterns, improving accuracy for complex, irregular handwriting.
Classification and Recognition	
Template matching based	Compares input characters to a stored set of templates; effective for standard fonts and known variations but struggles with variations in handwriting and font distortions.
Hidden Markov Model (HMM) based	Models the sequence and probabilistic structure of character strokes; effective for handling sequence-based recognition tasks but limited in capturing spatial features.
Support Vector Machine (SVM) based	Utilizes kernel functions for feature separation; effective for binary classification within a high-dimensional feature space, often used with extracted character features.
Neural Network/MLP based	Multilayer Perceptron (MLP) networks learn complex feature representations; effective in recognizing a variety of character patterns but sensitive to overfitting with limited data.
Fuzzy	Uses fuzzy logic for uncertainty handling, enabling the recognition of characters with unclear or ambiguous features; adaptable to character variations but complex to design.
Deep Learning based	Includes CNN, VGG, ResNet, LSTM models; excels at learning spatial and hierarchical features, enabling robust recognition of diverse handwriting styles. Computationally intensive but highly accurate.
Other Combined approach	Combines multiple techniques, such as hybrid SVM-HMM or CNN-LSTM models, genetic algorithms and other combinations, aims to leverage strengths of different methods for improved accuracy and robustness in complex cases.

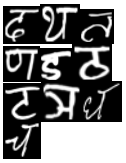
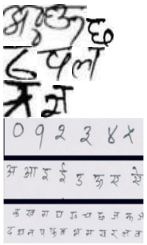
(b)

FIGURE 12. (a) General Pipeline of HDCR systems. The pipeline is: Input → Preprocessing → Segmentation → Feature Extraction → Classification and Recognition → Output (b) Models and techniques applied for Devanagari character recognition.

FIGURE 12. *(Continued.)* (a) General Pipeline of HDCR systems. The pipeline is: Input → Preprocessing → Segmentation → Feature Extraction → Classification and Recognition → Output (b) Models and techniques applied for Devanagari character recognition.

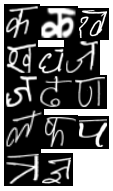
number of research papers referred across four categories, with the majority focusing on handwritten text, followed by general text, and fewer papers on printed and cursive text. Fig. 11 shows the number of references categorized by script, with the highest number for Devanagari, followed by Common OCR, and a smaller number for other languages. Here,

TABLE 1. Description of devanagari character datasets.

S. No.	Contributor name	Sample images	Description	Access Link
1	UCI Machine Learning Repository Shailesh Acharya and Prashnna Gyawali, 2016		Images containing 46 classes of characters and digits. Each class has 2000 examples. The dataset is split into a training set(85%) and a testing set(15%). The highest accuracy obtained on the test dataset is 98.47%	https://archive.ics.uci.edu/dataset/389/devanagari+handwritten+character+dataset
2	**Kaggle Devnagarai Character Data Set** Ashok kumar Pant, 2012		12,912 images containing images of three individual categories: Numerals (288 samples per class, 10 classes), Vowels (221 samples per class, 12 classes), and Consonants (205 samples per class, 36 classes).	https://www.kaggle.com/datasets/ashokpant/devanagari-character-dataset
3	Mendeley Handwritten Devnagari Character dataset Duddela Sai Prashanth, R Vasanth Kumar Mehta, Nagendra Panini Challa 2021		38,750 digitized images with 22,500 Devanagari Numerals (2250 each of 10 classes) and 16,250 Vowels (1250 each of 13 classes).	https://data.mendeley.com/datasets/pxrnvp4yy8/3
4	Computer Vision and Pattern Recognition (CVPR), Indian Statistical Institute (ISI) Kolkata, 2009		Approximately 30000 samples of handwritten isolated basic characters. of 49 possible classes. 22556 samples of Devanagari numerals collected from 1049 persons	**https://www.isical.ac.in/~ujjwal/download/database.html**
5	Centre for Pattern Recognition and Artificial Intelligence (CPAR) dataset Rajiv Kumar, Kiran Kumar et al 2019		CPAR contains 48 Characters and 9 Digits. It consists of 35000 images of numerals and 78400 images of characters.	**https://pypi.org/project/cpar/#description**

we want to acknowledge that some of the publications in this study are taken from sources other than the IEEE, Scopus, and Google Scholar databases. Finally, this study discusses around 200 papers.

TABLE 1. *(Continued.)* Description of devanagari character datasets.

6	Kaggle dataset, Modified National Institute of Standards and Technology (MNIST), 1994		Devanagari numerals dataset containing 20000 images. 1700 images of 0-9 class in train set and 300 images per class for test set	**https://www.kaggle.com/datasets/anurags397/hindi-mnist-data**
7	Kaggle Dataset Rishi Anand 2018		92,000 images corresponding to 46 characters, and the digits 0 to 9. The vowels are missing.	**https://www.kaggle.com/datasets/rishianand/devanagari-character-set**
8	Santosh K.C., INRIA Nancy Grand EST Research center, France, 2010		Online handwritten Devanagari characters are composed of 1800 samples from 36 character classes obtained by 25 native writers. Each writer was asked to provide two samples per class	**http://www.iapr-tc11.org/mediawiki/index.php/Devanagari_Character_Dataset**
9	Google code Devnagari database Vikas Dongre[36], 2012		5137 and 20305 isolated samples for numeral and character databases from 750 writers of all ages, sex, education, and professions. The offline sample images are stored in TIFF image	**https://code.google.com/archive/p/devnagari-database/downloads**

II. HANDWRITTEN DEVANAGARI CHARACTER RECOGNITION

Handwritten Devanagari Character Recognition (HDCR) systems follow a multi-stage pipeline to recognize handwritten characters accurately. The pipeline begins with image preprocessing, where the input image is prepared for analysis. This includes binarization, noise reduction, skew correction, and normalization. Next, segmentation isolates individual characters, a challenging task due to handwriting variations and touching characters. Once segmented,

feature extraction captures relevant statistical, geometrical, topological, or structural features for recognition. Finally, classification employs multiple machine learning algorithms, or deep learning algorithms, to identify characters based on extracted features. A general pipeline of HDCR and a list of models and techniques applied for Devanagari characters are shown in Figure 12 a) and b).

A. HANDWRITTEN DEVANAGARI CHARACTER DATASETS

There were several datasets available for handwritten Devanagari characters.

Researchers utilized publicly available datasets like the Devanagari Handwritten Character Dataset from the UCI Machine Learning Repository or Kaggle. These datasets often contain pre-processed images and labels. Many public datasets don't meet specific requirements, so researchers have considered creating a custom dataset. Devanagari handwritten characters were collected from online forums, crowdsourcing platforms, or by directly collecting data from individuals.

After creating data sets, image preprocessing techniques like normalization, noise reduction, greyscale conversion, and data augmentation techniques were used. Under data augmentation techniques, collected samples were used to create new samples with large data sets, and techniques like rotation, scaling, shearing, and flipping were used. Researchers ' use of various data augmentation techniques is discussed below in ***PREPROCESSING STEPS***.

However, the availability of datasets may change over time, and new datasets may become available. Here are a few datasets that were known then and are still available.

B. PREPROCESSING STEPS

The image acquisition process is crucial in addressing OCR challenges, as real-world images often exhibit various issues, such as noise, blur, and other quality degradations that may differ from the testing environment. In OCR, the recognition system acquires input images in specific formats like .jpeg or .bmp through scanners, digital cameras, or other digital input devices. In the context of Handwritten Devanagari characters, samples are collected from different individuals and scanned and converted into picture formats. Preprocessing is one of the essential steps in handwritten Devanagari character recognition, as the system's accuracy depends on how well the preprocessing has been performed. Preprocessing involves thresholding, thinning, noise reduction, size normalization, skew correction, and size normalization, as shown in Figure 13 below.

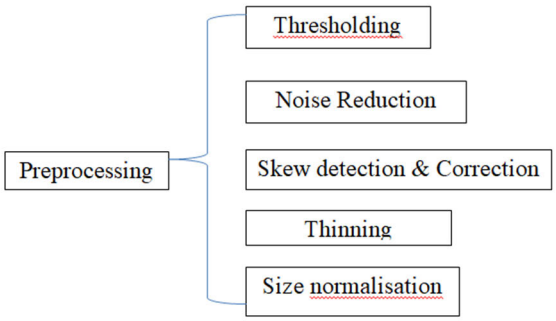


FIGURE 13. Preprocessing steps.

1) THRESHOLDING

Binarization or thresholding involves converting a grayscale image into a binary one. Two approaches exist for converting a gray-level image to binary: global and local thresholds [37]. Global threshold selects a single threshold value based on estimating the background level from the image's intensity histogram. Different values are chosen for each pixel based on local area information in local or adaptive thresholding. Binarization aims to identify object boundaries and emphasize shape analysis. Thresholding involves converting or adjusting binary shape images derived from processes like edge detection, boundary detection, or thresholding to have 1-pixel wide lines [38]. Various other thresholding methods [39], such as Sauvola, Niblack, Midgray, Bernsen, Mean, Median, Wolf thresholding, and Otsu, adaptive [40] thresholding methods are available in the literature.

2) NOISE REDUCTION

Image sensors often have inherent electro-mechanical limitations, leading to unavoidable noise during image acquisition. Addressing this issue at the preprocessing level becomes essential. One prevalent form of noise is isolated dots, resembling salt and pepper noise, which can significantly impact recognition accuracy. In a study by Prabhanjan et al. [41], diverse image-filtering techniques were utilized for noise removal. Linear filtering operations were applied to smooth images, followed by morphological opening and closing operations to eliminate small isolated components and bridge narrow openings in characters.

In another study [42], the authors employed a noise reduction technique based on Median Filter (MF) during the pre-processing stage of handwritten character recognition for Devanagari documents. The scanned handwritten document undergoes several pre-processing steps, including RGB to gray conversion, thresholding, complement of the image, morphological operations, linearization, and noise removal using MF. The Median Filter is a statistical approach for nonlinear signal processing that effectively reduces noise in the image. It replaces each pixel value with the median value within a defined neighborhood, which helps preserve edges and details while smoothing out unwanted noise. The authors used a specific formula to determine the noise variation for the Median Filter, and the results demonstrated the effectiveness of this noise reduction technique in enhancing the overall performance of the handwritten character recognition system.

3) SKEW DETECTION AND CORRECTION

Research Survey on Skew Detection of Devanagari Script [43] covers various skew detection techniques, mainly

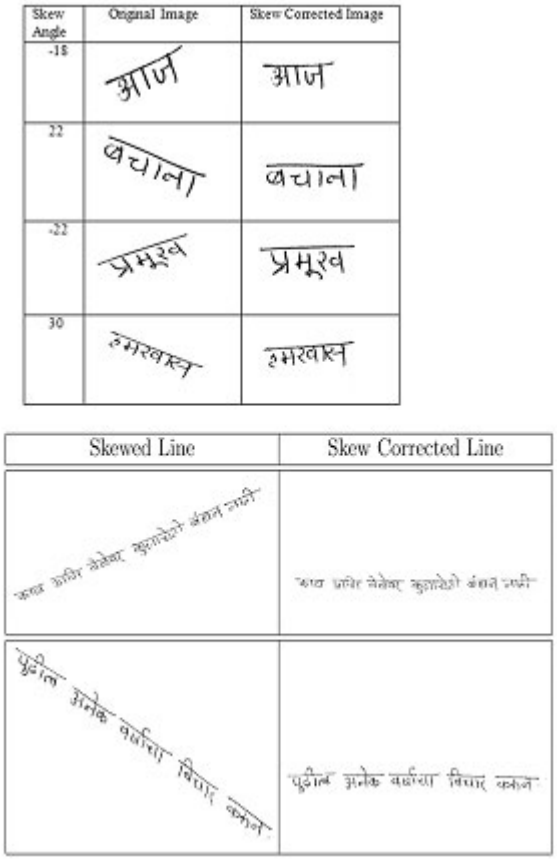


FIGURE 14. Sample result of skew-corrected words and lines [124].

applied to Devanagari script and other scripts. Methods such as the Hough transform [44], Fourier transform, projection profile, principal component analysis, nearest-neighbor clustering, interval halving [45], linear regression with axes parallel rectangle [46], and cross-correlation are explored in detail. Considering its unique writing system, Devanagari script skew detection challenges are highlighted. The paper also reviews the literature on skew correction methods, distinguishing between direct and indirect approaches.

Thapliya et al. [47] introduced a system with a skew angle detection feature to address mispositioning during scanning. Skew angles caused by misalignment are detected by rotating a standard reference image at different angles and cross-correlating it with an unknown letter. The peak signal identifies the optimal skew angle, and this information is utilized to correct the orientation of the characters during recognition.

Friends technique [48] introduces a novel approach for the segmentation and skew correction of handwritten Devanagari text. The skew detection involves tilting the handwritten text image at various angles, evaluating the height at each step, and selecting the angle with the minimum height as the final corrected image. The skew correction is further refined by considering the word's width and adjusting for residual skewness.

The proposed method in a paper by Guru et al. [49] for skew detection and correction in handwritten Devanagari word images employs morphological skeletonization, followed by a small eigenvalue-based line detection algorithm to extract linear segments from the edge image. These segments undergo clustering based on their orientations computed using eigenvectors corresponding to small eigenvalues. Connected small line segments within each cluster, determined by a predefined threshold, form longer lines. The Shirorekha, identified as the longest line among all clusters, serves as the baseline, with its orientation computed using endpoints. Line pixels are labeled, and K-means clustering is applied based on angles, facilitating the connection of disjoint lines. The orientation of each line pixel is determined using eigenvectors associated with small eigenvalues, and the Shirorekha line undergoes evaluation to address breaks caused by pixel removal. The skew angle is computed from the Shirorekha's endpoints, and the entire word image is skew-corrected. Experimental results on a dataset demonstrate the method's efficacy and superiority over the Hough transform-based approach in accuracy and efficiency.

Another paper [50] proposes a method for Skew angle detection of a cursive handwritten Devanagari script character image. It involves a heuristic approach based on the inherent dominating features of the script. The document is initially scanned horizontally and vertically to record the coordinates of the first information pixel encountered while simultaneously demarcating word boundaries. Ambiguities in the data are addressed by detecting line features and linear portions of curves. A model is then applied, considering weights assigned based on the number of ambiguities and the direction of the scan. The tilt angle is determined, and a transformation technique is employed for tilt removal. The method has been successfully applied to 235 writing samples, demonstrating accuracy and efficiency in skew detection and correction for the Devanagari script.

4) SIZE NORMALIZATION

Size normalization is crucial to prepare online input data for further processing. Rather than normalizing individual strokes, the approach involves normalizing entire characters. This is achieved by fitting the entire character into a 29×29 box using global minimum and maximum spatial coordinates across all strokes within the character [51].

Prabhanjan et al. [41] achieved uniform feature dimensions for classifiers, they resized images to a standard 40×40 size using bilinear interpolation. This ensured fixed feature vector lengths and simplified and optimized the feature extraction process based on empirical study findings. Size normalization in the proposed system [52] involves removing redundant points from the input character, followed by a scaling transformation to standardize horizontal and vertical dimensions. The normalized input is further smoothed by replacing each

point with the average of three consecutive points, repeated thrice for distortion elimination.

5) THINNING

Thinning enhances character visibility and structure in scanned images by reducing width. Thinning selectively removes foreground pixels, consistently extracting shape-related features from the characters [53]. Thinning enhances character visibility in scanned images by narrowing their width. Image Thinning is crucial in numeral recognition, but variations in stroke thickness due to different writers, pens, and image resolutions pose challenges. The algorithm proposed in the paper [54] adopts morphological thinning to ensure independence from these factors, reducing numerals to a uniform single-pixel width. Khanduja et al. [55] proposed a raster-based algorithm, demonstrating effectiveness in the Devanagari script character recognition.

In another paper [56], the proposed thinning algorithm uses a rotation-invariant method preserving character shape through rule-based thinning. Its rotation invariance ensures thinning to central lines, applying different rules concurrently for symmetrical thinning and enhanced system speed. A neural network-based approach is also applied to thinning [57].

6) DISCUSSION

Preprocessing steps are necessary to improve image quality before recognition. In our comparative analysis of image binarization as a pre-processing step for handwritten Devanagari scripts, we discovered that the Otsu method removes redundant and undesirable data from images while keeping only the needed information and applies even when the documents are deteriorated; the Otsu technique selects and applies the best threshold value. While Sauvola's and Niblack's approach eliminates background noise, it frequently results in severely thinned and broken characters. Wolf's method works well for most photos; however, if intensity variations are minimal, there are times when the characters break or disappear. The binary image produced by the Bernsen, Mean, and Midgray approaches has a lot of background noise, especially when the window is empty, and users must manually change the parameters. Thus, Otsu thresholding is superior for handwritten Devanagari characters overall.

In noise reduction for HDCR, filter-based techniques, morphological operations, adaptive filtering, frequency domain filters, median, and Gaussian filters are frequently employed. However, anisotropic diffusion, a combination of methods, and deep learning-based denoising models can also improve the quality of images before recognition. It is crucial to consider both the dataset's unique features and the type of noise present in the handwritten text when selecting a noise reduction method for HDCR. Additionally, the choice may depend on the trade-off between noise reduction and preserving essential details in the characters.

It is challenging to handle noise, differences in writing styles, and non-uniform backgrounds for skew detection and correction. The header line is crucial for detecting skew and correcting handwritten Devanagari characters. Handling skews in non-shirorekha characters or words can be a little challenging. Compared to other methods, the Hough transform method yields highly accurate results; nonetheless, it is computationally expensive and relatively slow. Additionally, the process is slow when there is noise. Also, the hough transform does not give good results for non-shirorekha characters or words. The output quality of the binarization process has a significant impact on the nearest neighboring method of skew correction. The projection profile method requires the calculation of the projection profile for every angle that can exist, which makes it inefficient. The skew detection accuracy depends on the angular resolution of the projection profile, and they can only estimate skew angles within $\pm 10°$ to $15°$. It is incredibly noise-sensitive. Because the 2D Fourier transform of every pixel in the document must be computed, the Fourier transform approach is computationally expensive for large images. The most significant density direction in Fourier space may frequently differ from the actual skew direction for a document image.

Size normalization helps the character to reduce the impact of variations during recognition. In addition to size normalization, adding different writing styles to the data enhances the model's capacity to generalize across diverse representations.

Ideally, The thinning algorithm should produce a perfect unit pixel-wide character skeleton without losing connectivity and information. When neural network-based thinning techniques are used, there is a noticeable improvement in the compression ratio and pixel removal parameter.

It's important to note that the efficacy of preprocessing methods can vary depending on particular needs and the type and extent of noise in the handwritten text. Several techniques and experimentation are often required to achieve optimal Handwritten Devanagari Character Recognition results. Additionally, the choice of a specific technique may depend on the specific requirements and constraints of the application.

C. SEGMENTATION

Segmentation is required to extract lines, words, and characters from text. Various researchers have focused on the segmentation of characters from the Devanagari script. This Devanagari script is handwritten or machine-written. It is effortless to segment the characters written using a machine. This machine-written text has uniform alignment and predictive writing. However, the segmentation results are reduced to substantially low for handwritten text. This handwritten text has a non-predictive nature. Each user writes the text in their own way. Also, the text written using the hand is not equally spaced. The segmentation level can be line, word, and character. The text in this can be broken and skewed. The technique used for extracting the characters from the Devanagari handwritten text is Collusion dilation. The header

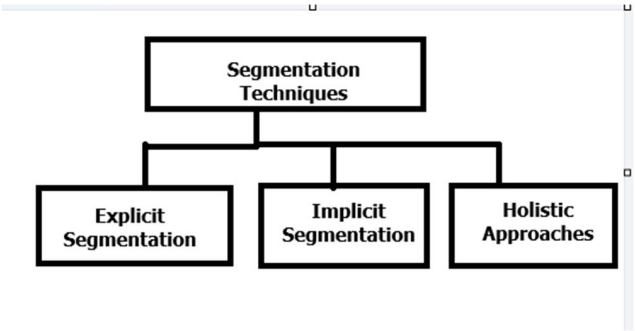


FIGURE 15. Categorization of segmentation techniques [59].

line is removed, and upper zone characters, middle zone then characters, and lower zone characters are recognized [58].

Choudray [59] broadly classified segmentation into three categories, namely Explicit Segmentation (Pure Segmentation), Implicit Segmentation (Recognition-based Segmentation), and Holistic (Segmentation Free) Approaches, as shown in Figure 15.

Explicit segmentation techniques used in literature are dissection-based segmentation techniques [60], locating minima in the upper contour of words, location of holes, contour direction, and core region position [61], hypergraph model [62], iterative cross-section sequence graph (ICSSG) [63], heuristics detection based technique [64].

Implicit segmentation (recognition-based segmentation), in which the system searches the image for components that match classes in its alphabet. In literature, the techniques used are the Hidden Markov model [65] and the two-stage HMM [66].

A holistic (Segmentation Free) process recognizes an entire word as a unit. They do not deal directly with letters but only with words; recognition is necessarily constrained to a specific lexicon of words. Very few researchers tried this technique.

As a conventional technique for text line segmentation, global horizontal projection analysis of black pixels has been utilized in [67] and [68]. Many researchers employ piecewise horizontal projection analysis of black pixels to segment text pages of different languages [69], [70]. The text-page image is decomposed into horizontal stripes using a piecewise horizontal projection technique. The positions of potential piecewise separating lines are obtained for each stripe using horizontal projection on each stripe. The potential separating lines are then connected to achieve complete separating lines for all respective text lines in the text page image.

Bansal et al. [71] 1999 presented the complete OCR Overview of Devanagari Scripts in their doctoral thesis. She did segmentation using the two-stage, hybrid approach. The initial segmentation process extracts the header line from the whole text and then extracts and separates the upper strip from the rest of the text. This results in vertically discrete character boxes that might be touching characters, conjuncts, lower modifiers, shadow characters, or a combination. Segmentation by this method is done based on structural information obtained from edge traversal in the second stage. After observing the coverage of the inner strip, horizontal zero crossing, vertical bar features, number and position of vertex points, moments, etc., the classification process is complete. She also does error detection and correction in her thesis under the post-processing phase. She reports an accuracy of 93% on character level as a performance measure [72]. The nonlinear Fuzzy approach is used for horizontal line estimation. They developed two fuzzy measures, horizontal feature and vertical feature, to identify the membership value of each pixel and its potential for belongingness to Matra. The horizontal feature is the corresponding horizontal longest-run component that exceeds the mean horizontalness of the respective words. It is used to extract header lines and horizontal elements of matras. The vertical feature is used to extract vertical stripes in word images. This verticalness property of the Matra may be removed from the column-wise count of a continuous run of black pixels. The fuzzy membership function for headline estimation is used, as shown in Figure 17.

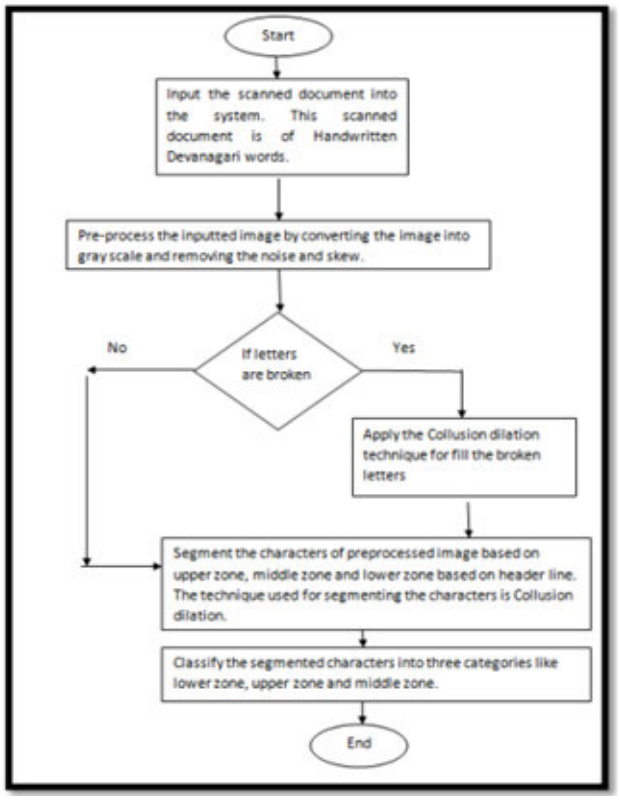


FIGURE 16. Flow chart of segmentation [58].

Mankar and Dongre [67] have used a histogram approach to extract lines, words, and characters. They counted numbers for black and white pixels, and using count, they identified boundaries and segmented them at three levels. Sharma et al. [73] 2006 proposed a technique that segments the word recursively by focusing on the presence of header

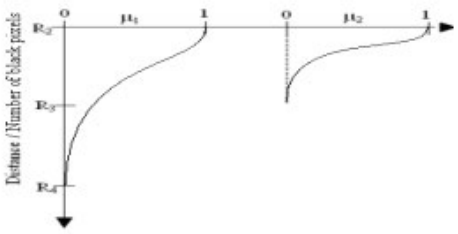


FIGURE 17. Fuzzy membership function [71].

lines, vertical and horizontal projection profiles, and the aspect ratio of characters. Handwritten text has the problem of overlapping, connecting half characters, and merging characters within a word more than that of typewritten text. He proposed an approach for segmentation that can be used for handwritten text and typed written text of Indian languages, viz. Devanagari, Bangla, Gurmukhi, etc., have structural features similar to the Gurmukhi script. Thakral et al. [74] 2014 represented a new technique for segmenting overlapping characters and conjuncts in the Devanagari script or Hindi language. The algorithm focuses on the Cluster Detection technique and works well for various input character sets. Handwritten Devnagari text recognition is a tedious task due to the different styles of writing of all persons, variations in handwriting, different font sizes of all characters, structural properties of language, and the presence of lower and upper modifiers. They tried to overcome all these challenges. So many of the issues mentioned above have been resolved with the help of a single algorithm that gives good results with different types of inputs.

The mentioned algorithm accurately segments the middle region of the Devanagari script on the word level. Peng et al. [75] discuss several segmentation-based recognition techniques for multilingual OCR. A novel method using three horizontal zones and linear regression [76] for segmentation has given high accuracies on self-generated datasets. A faster region-convolutional neural network (R-CNN) method uses residual and region proposal networks to effectively segment text lines in the Devanagari script [77]. Tested on a self-generated dataset, the proposed method achieved a 99.98% f-measure, outperforming existing state-of-the-art techniques.

Discussion: Segmentation is a crucial step in the Devanagari character recognition pipeline. The choice of segmentation method depends on the characteristics of the input data and the challenges the Devanagari script poses.

Thresholding and Binarization technique for segmentation is simple but effective, especially in scenarios with good contrast between characters and background. Connected Component Labeling helps separate characters based on connected components, assuming characters are well-isolated. Edge Detection techniques are effective in scenarios where characters have well-defined edges but may be noise-sensitive. Watershed Transform techniques help separate touching or overlapping characters but may require careful marker placement.

Hybrid approaches that combine traditional image processing techniques with deep learning methods show promise for handling the complexity and variability in Devanagari characters.

D. FEATURE EXTRACTION

Various feature extraction methods are classified into three major groups:

1. Global Transformation and Series Expansion
2. Geometrical and Topological / Structural Features
3. Statistical Features
4. Ensembled Feature Extraction Techniques

A good survey on feature extraction methods for character recognition can be found in [78].

1) GLOBAL TRANSFORMATION AND SERIES EXPANSION

A continuous signal generally contains more information than needs to be represented for classification. One way to describe a signal is by linearly combining a series of simple, well-defined functions. The linear combination of coefficients provides a compact encoding known as transformation or/and series expansion. Deformations like translation and rotation are invariant under global transformation and series expansion. Common transform and series expansion methods used in the CR field are [68]:

a: FOURIER TRANSFORMS

The general procedure is to choose the magnitude spectrum of the measurement vector as the features in an n-dimensional Euclidean space. One of the most attractive properties of the Fourier Transform is the ability to recognize the position-shifted characters when it observes the magnitude spectrum and ignores the phase [79]. Fourier Transforms has been applied to Devanagari OCR in many ways [80].

b: GABOR TRANSFORM

It is a variation of the windowed Fourier Transform. In this case, the window used is not a discrete size but is defined by a Gaussian function.

c: WAVELETS

Wavelet transformation is a series expansion technique that allows us to represent the signal at different levels of resolution. The segments of a document image, which may correspond to letters or words, are represented by wavelet coefficients corresponding to various levels of resolution. These coefficients are then fed to a classifier for recognition [81].

d: MOMENTS

Moments, such as central moments, Legendre moments, and Zernike moments, form a compact representation of the

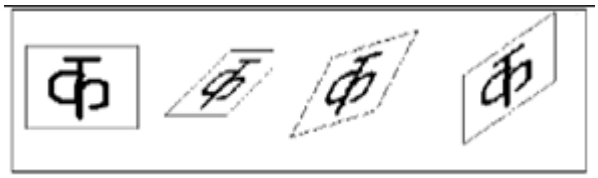


FIGURE 18. Moment invariant features of handwritten characters [68].

FIGURE 19. Different partitions of a devanagari character are used for computing features [45].

original document image that makes recognizing an object scale, translation, and rotation invariant [82], [83]. Moments are considered series expansion representations since the original image can be reconstructed entirely from the moment coefficients [68]. The moment invariant is an appropriate measure for tracing the noise-free image pattern. The features invariant to a character's scaling, translation, and rotation transformation help recognize many variations of the same character [68]. The 2-D moment of order (s + t) of a digital image f(x,y) of size m × n is defined as follows.

$$m_{st} = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} x^s y^t f(x,y)$$

where s = 0,1,2…and t = 0,1,2,…

e: ZERNIKE MOMENTS

Zernike moments are a class of orthogonal moments practical in image representation. The orthogonal property of Zernike polynomials enables the contribution of each moment to be unique and independent of information in an image. A Zernike moment does the mapping of an image onto a set of complex Zernike polynomials. Zernike Moments are defined over the unit disk instead of the actual plane and exhibit the orthogonal property [84].

Moment-based features are extracted from each zone of the scaled character bitmapped image. The image is partitioned into zones, and features are extracted from each zone. This paper proposes Zernike moments-based feature extraction for off-line Devanagari [84].

f: KARHUNEN-LOEVE EXPANSION

An eigenvector analysis attempts to reduce the dimension of the feature set by creating new features that are linear combinations of the original ones. It is the only optimal transform in terms of information compression. Karhunen-Loeve expansion is used in several pattern recognition problems, such as face recognition [68].

These methods can effectively capture distinctive shape characteristics, especially for characters with unique structures. They are helpful for characters with complex textures. Gabor is an example of a texture-based approach.

2) GEOMETRICAL AND TOPOLOGICAL / STRUCTURAL FEATURES

Structural features are based on the character's topological and geometric properties. Structural features include the number of horizontal lines, vertical lines, cross points, endpoints, horizontal curves at the top and bottom, etc. These features are then used to identify the shape or structure of the character [68]. Various topological and geometrical representations can be grouped into four categories:

a: EXTRACTING AND COUNTING TOPOLOGICAL STRUCTURES

In this category, lines, curves, splines, extreme points, maxima and minima, cups above and below a threshold, openings to the right, left, up and down, cross (X) points, branch (T) points, line ends (J), loops (O), direction of a stroke from a special point, inflection between two points, isolated dots, a bend between two points, horizontal curves at top or bottom, straight strokes between two points, ascending, descending and middle strokes and relations among the stroke that make up a character are considered as features [78], [85].

b: MEASURING AND APPROXIMATING THE GEOMETRICAL PROPERTIES

In this category, the characters are represented by the measurement of the geometrical quantities such as the ratio between width and height of the bounding box of a character, the relative distance between the last point and the last y-min, the relative horizontal and vertical distances between first and last points, distance between two points, comparative lengths between two strokes, width of a stroke, upper and lower masses of words word length curvature or change in the curvature [85], [86], [87].

Freeman's chain Coding is one of the most popular coding schemes. This coding is essentially obtained by mapping the strokes of a character into a 2-dimensional parameter space, which is made up of codes. There are many versions of chain coding. The character frame is divided into the left-right sliding window, and each region is coded by the chain code [88], [89]. Chain codes are a kind of directional code. The codes may be 4-directional or 8-directional depending upon the 4-connectivity or 8-connectivity of a pixel to its neighboring contour pixel. Chain code histogram-based features have been used by Kimura et al. [90] for hand-printed numeral recognition. All possible existing contours are traced, and their chain codes are produced using the Freeman chain code algorithm. As already mentioned, the codes may be 4-directional or 8-directional. The 4-directional code may be obtained from 8-directional codes where the contour pixels having codes 4, 5, 6, and 7 are assigned 0, 1, 2, and 3 codes, respectively. Contour-based Features are like the number of concavities, convexities, and the shape of the character's

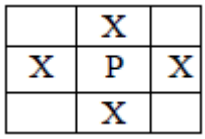


FIGURE 20. Contour point detection [91].

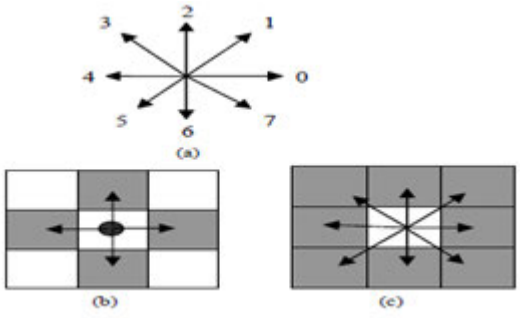


FIGURE 21. Chain Coding: (a) Direction of connectivity, (b) 4-connectivity, (c) 8-connectivity [91].

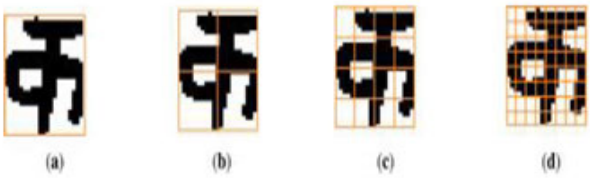


FIGURE 22. Contour point detection [44] a) One zone, b) 4 zones, c) 16 zones and d) 64 zones.

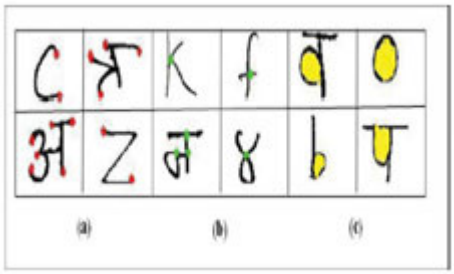


FIGURE 23. A character image: Endpoints, intersection points, loop [44].

boundary. Arora et al. [91] used chain code histograms of character contour. They considered a 3 × 3 window surrounded by the object points of the image. If any of the 4-connected neighbor points is a background point, then the object point (P), as shown in Figure 20, is considered a contour point.

Each contour pixel is assigned a different code that indicates the direction of the next pixel that belongs to the contour in some given direction. Chain code provides the points in relative position to one another, independent of the coordinate system. Chain codes are generated by detecting the direction of the next-in-line pixel [53]. In [92], directional chain codes of contour points of character images are used for character recognition. The contour points are the points of the image whose any of the four neighbors is the background pixel. After contour extraction, the image is divided into 7 × 7 blocks, and a chain code histogram is computed from each block. The obtained features are fed as input to the quadratic classifier for classification.

c: GRAPHS AND TREES

Words or characters are first partitioned into a set of topological primitives, such as strokes, holes, cross points, etc. Then, these primitives are represented using attributed or relational graphs. Image is defined either by graph coordinates of the character shape or by an abstract representation with nodes corresponding to the strokes and edges corresponding to the relationships between the strokes. Trees can also represent words or characters with a set of features with a hierarchical relation [93], [94].

3) STATISTICAL FEATURES

Statistical methods are used to identify statistical features of a character. The statistical features are derived from the statistical distribution of pixels. These are obtained from the collection of points representing the matrix of characters. These features can be easily detected as compared to structural features. Statistical features are less affected by noise or distortions than structural features. Compared to topological or structural features, these features can be easily obtained [95], [96]. This can include statistical measures like mean, standard deviation, etc., for each zone. To derive the enhanced features from the character image that can significantly capture the trend of foreground pixel distribution as a statistical measure in a processed image,

a: ZONING

The frame containing the character is divided into several overlapping or non-overlapping zones. The densities of the points or some features in different regions are analyzed [97]. Zoning is one of the most popular feature extraction methods. The character image is divided into a predefined number of zones, and a feature is computed for each zone. The frame containing the character is divided into several overlapping or non-overlapping zones, and the densities of object pixels in each zone are calculated. The density is calculated as the number of object pixels in each zone divided by the total number of pixels [98]. In this zoning method, four types of statistical features are extracted, namely, intersection and open endpoints, centroid, horizontal peak extent, and vertical peak extent, for Devanagari ancient character recognition [98].

This technique helps capture variations in different parts of characters, providing a localized view.

b: CROSSINGS AND DISTANCES

A popular statistical feature is the number of contour crossings by a line segment in a specified direction. The character frame is partitioned into a set of regions in various directions, and then the features of each region are extracted. Crossings-based methods have been used in [99] and [100] for hand-printed character recognition and are generally used to detect the number of strokes presented in a character along a particular path. If this path is along the rows, it is called a horizontal crossing; if it is along the columns, it is called a vertical crossing. The crossings can be considered as the number of transitions from black to white or from white to black pixels along a particular path. Crossings can be extracted from an original character and its skeleton. Kim et al. [99] used crossings in raw form, but Arica et al. [100] used the median of the black pixel runs in each scan line.

c: PROJECTIONS

Characters can be represented by projecting the pixel gray values onto lines in various directions. This representation creates a one-dimensional signal from a two-dimensional image, which can be used to describe the character image [101]. Projection histogram-based features are motivated by [102], where the author used these in a hardware-oriented OCR. Moreover, this technique is also used to detect orientation in a document page or segmenting a page into lines, words, and characters. An image is tracked along a path from a side to find the projection histograms, and the number of black pixels in that path is counted.

A Histogram of Oriented Gradients (HOG) effectively captures the local gradient information in an image. It breaks down the image into small regions and computes histograms of the gradients within each region. These features are derived from histograms of horizontal and vertical projections of black pixels in some particular character areas. They are extracted from the normalized image of the character to obtain normalized histograms of black pixels both on the X-axis and the Y-axis.

4) ENSEMBLED FEATURE EXTRACTION TECHNIQUES

Often, a combination of feature extraction and selection techniques yields better results. Experiment with different methods to find the optimal features for your specific Devanagari character recognition task. Many researchers combined structural, statistical, and transformation features to achieve an accuracy of 98%. Implementing and evaluating these techniques in combination with a suitable machine learning or deep learning model can enhance the accuracy and efficiency of Devanagari character recognition systems.

Arora et al. [91] have used the three feature sets in multiple classifier systems. Shadow features are extracted from scaled bitmap character images. Chain code histogram features are extracted by chain coding the contour points of the scaled character bitmapped image. Intersection features are extracted from a scaled, thinned, one-pixel-wide skeleton of the character image. Reference [102] used the profiles, histograms, crossings, zoning, and Kirsch directional edges-based features degrade the performance of chain code histograms, gradient, TDIST, NPW, and DT-based features when combined. When zoning, chain code histograms, gradient, TDIST, NPW, and distance transform-based features are combined, the performance in case of some combination is not affected much. Authors in [103] proposed character recognition techniques based on simple components such as curves, loops, lines, and dots in Devanagari characters. The characters are represented as a three-level hierarchy and bifurcated into simple components. These components act as feature vectors and are classified using multilayer perceptron (MLP) and radial basis function (RBF).

In [104] and [105] binary character images are represented as regular expressions. Authors segment the images using continuous 1s in columns and then rows. The image is scanned from left to right and top to bottom to find segments in horizontal and vertical directions. Each segment is labeled based on its connectivity with the segment in the next column and row. The generated regular expression of each character is stored as a feature and compared with test patterns using minimum edit distance for classification. This approach does not consider strokes in character. In [106], regular expressions are generated for different character strokes. Information about each segment's starting point, end point, length, column number, sequence number, and connectivity code is stored. If the difference between the starting points of two consecutive segments is less than the threshold, they are assigned the same sequence number and generate a stroke. This way, a regular expression identifies each stroke in a character. For test patterns, generated stroke expressions are matched with stored regular expressions using string matching algorithms, namely minimum edit distance. Although this method uses different and new techniques for character recognition, it is highly prone to variations in writing styles. A small change in character shape can result in a significant shift in regular expressions. Authors in [107] explore topographic features, namely closed region, the convexity of strokes, and straight-line strokes from different directions: north, south, east, and west. A database containing different convex shapes is created to determine the convexity of strokes. A few convex shapes are shown in Fig. 24. The thinned image is scanned vertically and horizontally for all four directions to determine the set of four-connected pixels with value ≥ 5. The detected set of pixels matches the defined convex shapes of the database. These detected convex shapes form a topographic feature set. The closed regions are identified using connected component analysis; if the stroke pixels are eight connected, they create a closed region. If >20 pixels are horizontally eight connected, then those pixels form a straight line. After determining all these topographic features, undirected graphs are created where vertices are defined using the centroid of each topographic component, and edges are added with respect to their topology in a thinned character image.

FIGURE 24. Different convex shapes [58].

Discussion: Effective feature extraction is vital in Devanagari character recognition, providing a foundation for accurate classification. Combining traditional computer vision techniques with deep learning approaches can yield robust and efficient recognition systems. Regular evaluation and adaptation of feature extraction methods based on the specific challenges of Devanagari script recognition are crucial for achieving optimal performance.

Zoning and segmentation help to handle the complex script characteristics of Devanagari. Applying connected component analysis, skeletonization can effectively extract the core structure of the character, aiding in the separation of overlapped and touching characters. HOG techniques are effective in capturing local texture and shape information. It is beneficial for detecting edges and shapes within characters. Local Binary Pattern (LBP) techniques are robust to variations in illumination and capture textural information in characters. They are suitable for recognizing patterns in different parts of characters. Using raw pixel intensity values as features is a simple yet effective technique. It captures basic information about shapes, contours, and contrasts in characters. It is beneficial for straightforward character structures. Extracting geometric and topological properties (Shape Descriptors) of characters such as Fourier descriptors or Hu moments effectively captures distinctive shape characteristics. They are suitable for characters with unique and recognizable structures. Texture Descriptors describe the spatial arrangement of pixel intensities to capture texture information. They are helpful for characters with complex textures. For instance, Gabor filters can capture local texture patterns and global information.

CNN techniques automatically utilize convolutional and pooling layers to learn hierarchical features from input images. They effectively capture complex hierarchical features, especially when trained on large datasets. Suitable for handling variations and intricate structures in Devanagari characters. Adaptive and dynamic feature extraction parameters dynamically based on the characteristics of the input data enhance adaptability to varying writing styles and image variations. Helpful in handling diverse datasets and real-world variability. Hybrid feature extraction techniques can be used to capture diverse aspects.

E. CLASSIFICATION AND RECOGNITION

The development of handwritten Devanagari character identification has been ongoing for several decades; research efforts peaked in the late 20th century and continued into the 21st. Sethi and Chatterjee [108] released the first research study on Devanagari characters in 1976. The authors used a systematic method to identify the handwritten Devanagari numerals by determining the presence and locations of D-curve, C-curve, left slants, and horizontal and vertical line segments. Sinha and Mahabala [109] employed embedded picture language for the Devanagari script. A syntactic pattern analysis system with an embedded picture language has been designed for this purpose. The system stores structural descriptions for each script symbol regarding primitives and their relationships. The input word is digitized, cleaned, thinned, segmented (to extract composite characters), and labeled (a local feature extraction process). Using the stored description as a guide, recognition entails looking for primitives on the labeled pattern. Contextual limitations are also employed to determine accurate interpretation. In 1987, RMK Sinha stated in his paper that context plays a crucial role in interpreting Devanagari text, especially handwritten forms [110]. Correctly recognizing upper and lower modifier symbols is challenging due to improper placement, intermingling, or incorrect segmentation. To achieve a syntactically meaningful word, a Devanagari text recognizer must appropriately associate these modifier symbols with the core strip characters, making necessary substitutions. This paper outlines a method and shares findings on recognizing Devanagari text, incorporating script composition contextual information to post-process the identified symbols.

In 1995, Sinha and Bansal discussed the Devanagari document processing system, which utilizes diverse knowledge sources across all stages [111]. The first step in preprocessing is to extract a test zone from a document using a syntactic representation of document layout knowledge. Following this, the test zone undergoes segmentation into lines, lines into words, and words into characters. Owing to the complex structure of Devanagari characters, which comprise several symbols, specific algorithms are employed to divide characters into individual symbols instead of seeing them as one entity. Recognition of these symbols is based on various features extracted and stored during training. The identified symbols are reconstructed and subjected to validation using a partitioned dictionary. Thapliya et al. applied Joint Transform Correlator Recognition of the Devanagari Script [47]. They proposed a hybrid optoelectronic character recognition system for identifying Devanagari scripts applicable to old Sanskrit manuscripts. The system utilizes a joint-transform correlator (JTC) for correlation operations and electronic peripheral hardware for data transfer. The optical system design incorporates a liquid crystal spatial light modulator and a CCD camera. Through a one-channel JTC, the system successfully discriminates against all core elements, marking a notable achievement. Additionally, the paper explores novel techniques, including syllable partitioning and skew angle correction, leveraging unique characteristics of the script.

In 1996, Keeni et al. proposed a method for recognizing Devanagari characters using Neural networks [112]. The proposed method aimed to reduce the number of output units required in a conventional neural network, mainly relying

on a winner-take-all classification approach. It introduced an automatic coding procedure for representing the output layer and suggested an alternative method for final classification. The paper also outlines a heuristic approach for representing output units by leveraging the structural information of Devanagari characters. The study demonstrates that the representation of the output layer influences the generalization/performance of the network, as evidenced by random representations. The automatic representation method achieves a high recognition rate of 98.09% for 44 categories. Chaudhuri and Pal's paper [113] introduced an OCR system specifically developed to recognize two major Indian language scripts: Bangla and Devanagari (Hindi), originating from the ancient Brahmi script. The model covers various processes, including document digitization, skew detection, text line segmentation, zone separation, and word and character segmentation, employing the same algorithms for both scripts. The uniform approach extends to character grouping into basic, modifier, and compound categories. While there are differences in the feature sets, classification tree, and knowledge base for error correction between Bangla and Devanagari, the system demonstrates robust performance for single-font scripts on clear documents. Later, in 1999, Pal and Chaudhari also presented a paper on the automatic separation of machine-printed and hand-written text lines [114]. The paper addresses the challenge of dealing with documents containing a mix of machine-printed and hand-written texts, requiring distinct OCR methodologies for each. The proposed scheme for Bangla and Devanagari characters relies on structural and statistical features of text lines to effectively separate machine-printed and hand-written text before feeding them into the respective OCR systems. The classification scheme demonstrates an impressive accuracy of approximately 98.3%.

Again, in 1999, Sinha and Bansal introduced a framework for describing the shapes of Devanagari characters and their application in character recognition [115]. By leveraging specific script features, the method reduces the search space and establishes a reference for correspondence during matching. Description prototypes are created from segmented real-life scripts, accounting for aberrations introduced during segmentation. When applied to printed Devanagari text, this method attained an accuracy of around 70% without post-processing, which increased to 88% with the assistance of a word dictionary for recognition. In 2000, they [71] published their research on Integrating knowledge sources in the Devanagari text recognition system. The Devanagari document recognition system described in this work is rooted in the understanding that knowledge in various forms and at different levels is integral to the reading process. The system relies on statistical learning and a specialized word dictionary tailored for optical character recognition (OCR) without engaging in reasoning on these sources. The study explores the relative importance and hierarchy of these knowledge sources. Some are obtained through automated training processes, while others are extracted from the processed text. The complete Devanagari OCR system has been developed and tested on real-life printed documents of diverse sizes and fonts, primarily photocopies. The system demonstrates a commendable performance, achieving approximately 90% correct recognition.

The first study on Recognition of Unconstrained On-Line Devanagari Characters was reported in 2000 [116]. Recognizing unconstrained Devanagari writing presents more challenges than English cursive writing, given variations in stroke order, number, direction, and shape. The online pen computing environment proves beneficial as a user-friendly interface for handling this complex script. In a character recognition experiment involving 20 different writers, each contributing 5 samples per character in an entirely unconstrained manner, a high accuracy of 86.5% with no rejects was achieved. This success resulted from integrating multiple classifiers that focused on either local on-line properties or global off-line properties.

Researchers started investigating various structural, statistical, and template-based pattern recognition strategies [117] and feature extraction combined with simple classification methods. Hidden Markov Models (HMMs), Support vector machines (SVM), Multilayer perceptron (MP), and other machine learning and soft computing techniques were explored for recognizing characters. One of the most advanced machine learning methods, deep learning, was first presented in 1998 for character identification using the MNIST database [118]. In essence, deep learning methods consist of several hidden layers, each with several neurons that calculate the appropriate weights for the deep network. These weights require a significant amount of processing power, which made it difficult for a strong system to come by at the time. Since then, by transforming the images into feature vectors, the researchers have focused on developing a method that requires less power. Deep learning methods began to emerge after 2010. A deep learning-based machine translation strategy has also been discovered for the Sanskrit-Hindi pair [119].

In the literature on Devanagari character recognition, common techniques for splitting data into training, validation, and test sets include random, proportion-based splits (such as 70% training, 15% validation, and 15% testing) and stratified sampling to maintain class balance. This ensures that every subset accurately reflects the entire dataset. To preserve the same class distribution in every subset, they employed stratified sampling if there was a class imbalance (for example, some characters are more common than others). K-Fold cross-validation is often used for dependable performance evaluation, usually with five or ten folds. Stratified K-Fold is sometimes used in studies to guarantee balanced class representation inside each fold. They divided the dataset into K equal-sized folds, trained the model on K-1 folds, and evaluated it on the remaining fold. This is repeated for K times, using a different fold for evaluation each time, and

TABLE 2. Structural and statistical approaches based on work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	Data Set	Gap Analysis
Abhiram Ranade et al. [120]	Online character recognition	2001	Pen direction	✗	✗	Unknown	Distortion of handwriting well captured with stroke and precomputation.
U Pal, BB Chaudhary. [121]	Offline recognition	2001	Structural and statistical features Headline, standard deviation of components	✗	98.6%	600 handwritten documents	Good separation of machine-printed and hand-printed lines
V Bansal and RMK Sinha [122]	Offline character recognition	2001	Devanagari text segmentation, utilizing filters	Distance-based classifier	93%	Unknown	Language variations cannot be handled
U Pal and N Tripathy [123]	Offline character recognition	2005	water reservoir concept-based character segmentation, contour distance	✗	97.3%	Unknown	Inability to segment multi-touching characters.
PS Deshpande, et al. [104]	Offline character recognition	2006	character encoding and regular expressions for shape	Regular expression	90%,	Unknown	Inefficient in handling overlapping cases and shape variations
N Sharma et al. [92, 124]	Offline character recognition	2006, 2007	directional chain code, information on contour points	quadratic classifier	98.86 % for numerals, 80.36 % for characters	Unknown	Similar shapes character confusion rate is high
PS Deshpande et al. [106]	Offline character recognition	2008	Regular Expressions	Minimum Edit Distance Classifier	82%	Own created 5000 samples	Not able to utilize larger and extensive data sample
R. J. Ramteke et al. [125]	Offline character recognition	2010	Invariant Moments, vertical, horizontal projection, Shirorekha-based segmentation	Fuzzy Gaussian Membership function	94.6%.	Unknown	Lacks details on clustering results
M Jangid [126]	Offline character recognition	2011	recursive subdivisions, zone density, and directional distribution	✗	94.89 %	Own data set of 12240 images	Testing the methodology's robustness on diverse datasets
L Malik et al. [127]	Offline character recognition	2012	subgraphs homeomorphic to predefined word prototypes	graph-based holistic approach	89%	Own database: 150 document images, testing: 1000, training: 1500 words	Explores improved global layout utilization
A Deshmukh, et al. [128]	Offline character recognition	2014	Principal Component Analysis and Eigenspace features	Gerschgorin's theorem	characters with modifiers perform well	Unknown	Challenges in recognizing fused characters
S Gaikwad, et al [129]	Offline handwritten Devanagari characters	2023	Entropy-based skew correction, Mask-based approach, HOG features	AdaBoost ensemble boosting method.	98.43 % and 98.68 % on V2DMDCHAR and ISIDCHAR databases	Table-1:data set-[4]	Emphasized the importance of pre-processing, specifically skew correction and

TABLE 2. *(Continued.)* **Structural and statistical approaches based on work done for handwritten devanagari characters.**

							header removal
KC Santosh, et al [130]	Online handwritten Devanagari character	2012	Spatial similarity-based stroke clustering with dynamic time warping (DTW)	✗	95%	36 characters from 25 speakers.	Future work involves extending recognition to the syllable level.

then the performance metrics are averaged across all K folds to obtain a reliable estimate of the model's performance. For enhanced robustness, Monte Carlo cross-validation is occasionally employed, involving multiple random splits to capture performance variability. In smaller datasets, Leave-One-Out Cross-Validation (LOOCV) provides a thorough evaluation but is less feasible for larger datasets due to computational demands. Hybrid approaches, such as combining an initial hold-out split with K-Fold cross-validation on the training set, are also observed to balance validation thoroughness with computational efficiency. These techniques help ensure robust and generalizable model performance for handwritten Devanagari character recognition.

Many have used random sample multiple training and validation sets from the original dataset. They trained and evaluated the model on each sampled dataset. It is estimated by averaging the performance metrics across all iterations.

1) STRUCTURAL AND SYNTAX BASED APPROACHES

These methods make use of the statistical patterns and character structure. Based on their geometric and structural characteristics, characters are studied. Characters must be dissected into their individual strokes, stroke angles, loops, pixel densities, distributions, and other structural components. Rules or models are constructed that indicate the predicted structural qualities of each character. For instance, two diagonal strokes meeting at the upper center could represent the letter 'A.' Characters with distinct geometric shapes can be recognized very well with structural approaches. Researchers' work on techniques based on syntax or structure is included in Table 2.

The pen direction method effectively captures stroke direction and pen precomputation. Structural and statistical feature methods achieved a high accuracy (98.6%). Devanagari text segmentation method uses a distance-based classifier and achieves an accuracy of 93%. Water reservoir, concept-based character segmentation, and contour distances give 97.3% accuracy. Regular expressions give 90% accuracy. Offline Handwritten Character Recognition with various techniques like contour information and directional chain code achieves an accuracy of 80.36% to 98.86%. Regular Expressions with minimum edit classifier achieve an accuracy of 82% but may struggle with overlapping characters and shape variations. Invariant Moments for character recognition uses a fuzzy Gaussian membership function classifier and achieves an accuracy of 94.56%. Recursive Subdivisions for character recognition achieve an accuracy of 94.89%. Subgraph Homeomorphic for character recognition uses a graph-based holistic approach and achieves an accuracy of 89%. Principal Component Analysis (PCA) and Eigenspace features for character recognition highlight the challenges of recognizing fused characters, but no information has been provided about their accuracy. Entropy-based skew correction with a Mask-based approach and HOG features achieve an accuracy of 98.43% to 98.68%. Spatial similarity-based stroke clustering with dynamic time warping (DTW) is 95% accurate.

Discussion: Structural-based approaches for recognizing handwritten Devanagari characters involve analyzing the structural components and relationships within the characters. These methods focus on capturing the characters' inherent geometric and topological properties to improve recognition accuracy. In the early years, researchers started using this approach. However, these techniques did not give appropriate results for all character sets of Devanagari and could not handle complex character structures. Stroke-based Methods are sensitive to variation in writing styles. The zone-based technique works on the definition of a zone. The contour-based technique is sensitive to noise. The effectiveness of these methods depends on the ability to capture and utilize relevant structural information while addressing challenges such as variations in writing styles and noise in the input data. Combining structural-based approaches with other techniques, such as statistical or machine learning, provides a more robust solution for Devanagari character recognition.

2) TEMPLATE-MATCHING BASED APPROACHES

The template refers to a specific character representation, and the matching involves comparing this template with different regions of the input image. One possible method for implementing a template-matching algorithm is to move the template over the input image and determine how similar it is at each location. By establishing a threshold, similarity can be calculated. When the similarity between the template and a region of the input image exceeds this threshold, consider it a match. Template-matching has limitations, particularly when accommodating differences in handwriting sizes, styles, and orientations. Additionally, several scholars worked on template-matching methods. Table 3 summarizes the work done for handwritten Devnagari characters.

The method achieves 97.29% accuracy in online character recognition through robust stroke feature mapping, utilizing human motor functionality and neuromotor characteristics. Applicable to alphabet-based languages, it supports word-level contextual details and can extend beyond its initial online dataset. Script identification, character segmentation, and recognition employ GHIC with 87.82% accuracy

TABLE 3. Template-matching-based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	DataSet	Gap Analysis
U Garain et al. [131]	Online Character Recognition	2002	Human motor functionality, neuromotor characteristics, pen movement	$ Robust stroke feature mapping	$ 97.29 %	Online data	This can be applied to other alphabet-based languages, followed by the incorporation of word-level contextual details.
H Ma et al. [132]	Script identification	2003	Three-stage work: script identification, character segmentation, and then recognition	GHIC (Generalized Hausdorff Image Comparison)	87.82 %, with an accuracy of 95% for ideal images	Hindi - English bilingual dictionary and Hindi documents in PDF	Lacks implementation details on the actual OCR correction or ambiguity resolution methods.
VK Nandury [133]	Offline character recognition	2005	Stroke-based features	Decision trees	65%.	Unknown	Performance enhancement needed
PS Deshpande et al [134]	Offline character recognition	2007	Encoding string	Regular expression	89%	3400 own-created samples	Better preprocessing and modifier processing needed
V Mane and L Ragha [93]	Offline vowels recognition	2009	Eigen-deformations estimation through principal component analysis	elastic image matching (EM) technique	94.91 %	ETL6 for English and own dataset of 3600 for Hindi	Reducing the displacement vector size, including consonants, also
L Malik et al. [135]	Offline character recognition	2010	Segmentation in words, regular expression	float-fill algorithm	95%	Unknown	Absence of standard dataset

(95% for ideal images), lacking OCR correction details. Offline recognition with added decision trees classifier with structural properties yields 65% accuracy, emphasizing the need for enhancement. Regular expression encoding achieves 89% accuracy on 3400 samples, highlighting preprocessing improvements. Eigen-deformations through principal component analysis and EM achieve 94.91% accuracy for offline vowels. While segmentation in words attains 95% accuracy, dataset details remain undisclosed, and the absence of a standard dataset underscores the need for benchmarking.

Discussion: Template matching is conceptually simple and easy to implement as it involves comparing a template (a small image or pattern) with different regions of a larger image to find instances where the template matches well. This technique was popularly used in earlier HDCR works. As handwritten Devanagari characters have wide variations, finding a single template that can accurately represent all variations becomes challenging. For each character, deciding whether to use a single or multiple templates representing variations in writing style is crucial. The computational efficiency of the template matching algorithm, especially when dealing with large datasets, is also of significant concern. Template matching relies on pixel-wise similarity, which can not effectively capture the subtle differences between similar characters. Discriminative features that distinguish one character from another are not explicitly learned, making it less robust in handling character variations. Adaptive template creation techniques, where templates are dynamically adjusted based on the input data, improve recognition accuracy. Combining template matching with other methods, such as machine learning classifiers, neural networks, and deep learning techniques, leverages both approaches' benefits.

3) HIDDEN MARKOV MODEL (HMM) BASED APPROACHES

A Markov chain with only a partially observable state is called an HMM. A hidden Markov model is a double stochastic process that presents as a grouping of perceptions through another arrangement of stochastic procedures. The underlying stochastic process in the hidden Markov model is not detectable. The hidden process consists of a set of states connected via transitions with probabilities, while the seen process consists of a set of outputs or observations, each of which may be emitted by any state in line with some output probability density function (PDF). The most likely outcome can be found by employing classification methods such as the Viterbi Algorithm or the HMM counter-algorithms, which are based on the observation chain and can be used to calculate the hidden states chain.

TABLE 4. Hidden markov model(HMM) based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	Data Set	Gap Analysis
PS Natarajan, et al [136]	Offline character recognition	2005	Lexicon/statistical grammar	Forward-Backward algorithm	Character error rate is 1.0-1.4% on clean data	Synthetic data: 800 different characters	Enhancement for optimization and degraded data.
U. Bhattacharya et al. [137]	Offline character recognition	2006	Directional-view strokes.	ANN	92.83 %	Table-1:dataset-[4]	Incorporate size and positional details of strokes to enhance the recognition rate.
SK Parui and B Shaw [138]	Offline character recognition	2007	Stroke primitives in a word image	✗	82.89 %.	Self-created: training-7000, testing-3000 word images	combining global and local approaches to increase efficiency
B Shaw, et al. [139]	Offline character recognition	2008	Chain-code directions histograms in image strips scanned by a sliding window	✗	80.2%	Self-developed: 100 words, 22,500 training, 17,200 test	Hybrid solution of segmentation-based and segmentation-free approach and post-processing
B Shaw and SK Parui [140]	Offline character recognition	2010	Two-stage schema	Modified Bayes discriminant function	91.25 %	Self-created: training-7000, testing-3000 word images	Explore large lexicon and other classifiers and feature
MA Shaikh and MR Dagade [141]	Offline character recognition	.2015	Re-ranking for offline handwritten Devanagari word recognition.	✗	83.77 %	2000 scanned images of Hindi City Names	Extending the system for entire documents
R Ghosh and PP Roy [142]	Online character recognition	2016	Whole-stroke, local structural features, and dominant point-based features	✗	93.82 %	350 different words	Combining zone-wise features and extending the applicability to other Indian scripts
R Ghosh and PP Roy [143]	Online handwritten Devanagari signature recognition	2017	Zone-wise Slopes of Dominant Points (ZSDP)	SVM	SVM with 16 zones, CT = 2, and linear kernel and HMM with 32 Gaussian mixtures and 3 states achieved the best results	500 samples of signatures	Further research in Indian scripts.

Lexicon statistical method with a forward-backward algorithm has an accuracy of 98.6% to 99% on clean data but lower accuracy on degraded data. Directional-view strokes method with an Artificial Neural Network (ANN) classifier achieves an accuracy of 92.83%. The stroke primitives method achieves an accuracy of 82.89%. The chain-code directions method with a histogram achieves an accuracy of 80.2%. The two-stage schema method with a modified Bayes discriminant function classifier achieves an accuracy of 91.25%. The re-ranking method achieves an accuracy of 83.77%. Whole-stroke, local structural features method combined with dominant point-based features achieves an accuracy of 93.82%. The zone-wise slopes of dominant points (ZSDP) method with SVM and HMM classifiers achieves an unspecified accuracy.

Discussion: HMMs are statistical models representing a system with hidden states, observable symbols, and transition probabilities between states. In the context of HDCR, HMMs can be used to model the temporal sequence of features in an HDC. HMMs allow for the modeling of state transitions,

TABLE 5. Support vector machine (SVM) based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	Data Set	Gap Analysis
CV Jawahar et al. [144]	Offline character recognition	2003	Principal Component Analysis (PCA)	✗	96.7 %,	200000 sample characters	There are no considerations for geometric or shape features.
U Pal et al. [145]	Offline character recognition	2008	Gradient and curvature	Modified Quadratic Discriminant Function	95.13%	only basic characters	Exploring additional fusion methods
S Arora et al. [146]	Offline character recognition	2010	Shadow, chain code, view-based, longest run	Neural Networks	SVM - 92.38% NN-93.93%	Table-1:dataset-[4] and 2254 samples own created	SVM can be tested on other scripts and numerals, also
B Singh et al. [147]	Offline character recognition	2011	Curvelet Transform for feature extraction	k-NN classifier	93.21%	Unknown	Poor-quality documents and noise effects are suggested
M Jangid, et al. [148]	Offline character recognition	2011	Recursive subdivision of the character image	✗	98.98%	Table-1:dataset-[4]	Confusion between certain numerals is reported
Jangid M, et al. [149]	Offline character recognition	2011	324 features utilizing density and directional distribution for zones	RBF kernel	98.94%	Table-1:dataset-[4]	Combining multiple classifiers and diverse features for improvement.
M Jangid and S Srivastava [150]	Offline character recognition	2014	Surpassing HOG and SIFT	GLAC (Gradient Local Auto Correlation)	93.21% - 95.21%	Table-1:dataset-[4] and Table-1:dataset-[9]	Similar-shaped character differentiation
A Gaur and S Yadav [151]	Offline character recognition	2015	Euclidean distance	region-based k-means clustering	95.86%		Exploration of classification methods
Ghosh R, Roy PP [152]	Online character recognition	2015	Zone-based ZSD, ZSDP	✗	92.48%	10,000 characters	Explore more features and classifiers
N Tripathy, et al. [153]	Offline character recognition	2016	PCA, circular zones, centroids, and angular information	✗	99.11%	7515 character images	Errors due to similar shapes
PK Singh, et. al. [154]	Offline numeral recognition.	2016	196-dimensional feature vector, including regional Weighted Run Length features	✗	95.02%	6000 samples of digits	Address misclassification errors, integrate global features, and expand the database.
SR Narang, et al. [155]	Offline character recognition	2018	RBF kernel, DCT zigzag features of length 100, Histogram of Oriented Gradients (HOG)	Naïve Bayes, Decision Tree	90.70%	5484 characters ancient manuscripts	Incorporating modifiers and conjuncts exploring more efficient features, additional classification techniques
SP Deore and A Pravin [156]	Offline character recognition	2019	Histogram of Oriented Gradients (HOG)	K-NN, Neural Network	87.38%	10560 character images	Increasing the database size, exploring

TABLE 5. *(Continued.)* **Support vector machine (SVM) based approaches work done for handwritten devanagari characters.**

							prominent features, dividing images into zones, and concavity analysis
SR Narang, MK Jindal and M Kumar [157]	Offline character recognition	2019	AdaBoost and Bagging methodologies with Discrete Cosine Transform (DCT) zigzag features	Decision tree, Naïve Bayes	91.70%	5484 pre-segmented characters of ancient documents	Incorporating modifiers and conjuncts exploring more efficient features, classification techniques
Kumar A et al. [158]		2024	morphological operations, edge detection, HOG feature extraction	Bagged trees, K-NN, ESD	96.77%	Unknown	Improve method to increase accuracy and recognize a word

representing the sequential nature of character writing. This is particularly beneficial in languages like Devanagari, where characters are often written in a specific stroke order. The success of HMMs in HDCR heavily relies on the choice of features. Effective feature extraction methods, such as those capturing shape, size, and stroke information, are crucial for accurately modeling the handwriting patterns of Devanagari characters. While HMMs can model temporal dependencies, modeling contextual information between characters, ligatures, and diacritics in the Devanagari script is non-trivial. HMMs are sensitive to the initial parameterization. Accuracy achieved using HMM methods is not promising for HDCR. HMMs are generative models that focus on modeling the entire data distribution. In character recognition tasks, discriminative models that explicitly learn decision boundaries between classes outperform HMMs, especially when dealing with complex Devanagari character variations.

4) SUPPORT VECTOR MACHINE (SVM) BASED APPROACHES

Within Devanagari, handwritten character recognition (HWR) and Support Vector Machines (SVMs) have emerged as prominent techniques due to their robust classification capabilities. Their effectiveness stems from their ability to identify optimal hyperplanes in high-dimensional feature spaces, effectively separating distinct character classes. This involves meticulous analysis of feature vectors extracted from handwritten characters, such as directional features or edge maps. By strategically positioning the hyperplane to maximize the margin between classes, SVMs enable the accurate classification of novel characters based on their proximity to established decision boundaries. This efficacy in handling Devanagari's inherent complexities, such as intricate ligatures and head strokes, underscores the suitability of SVMs for developing reliable and efficient HWR systems.

Principal Component Analysis (PCA) achieves 96.7% accuracy. This method doesn't consider the geometric or shape features of characters. Gradient and curvature with modified Quadratic Discriminant Function classifier achieve 95.13% accuracy. Shadow, chain code, view-based, longest run with Neural Networks (NN), and SVM classifier achieve 93.93% and 92.38% accuracy, respectively. Curvelet Transform with k-NN classifier achieves 93.21% accuracy. Using a recursive subdivision of the character image leads to a high error rate (98.98%) due to confusion between similar numerals. Using 324 features with RBF kernel achieves an accuracy of 98.94%. HOG and GLAC methods are compared for feature extraction. GLAC achieves a lower error rate (within 93.21% and 95.21%) than HOG (Surpassing HOG). Using Euclidean distance and region-based k-means clustering leads to an error rate of 95.86%. PCA achieves the lowest error rate (99.11%) among the methods listed. Using a 196-dimensional feature vector reduces the error rate to 95.02%. RBF kernel, DCT features, and Naive Bayes classifier lead to a high error rate (90.70%). Using HOG features and the K-NN classifier leads to a high error rate (87.38%).AdaBoost and Bagging methodologies achieve an error rate of 91.70%.

Discussion: SVMs are a powerful and widely used tool for handwritten Devanagari character recognition. Much work has been reported on HDCR using SVM. Their ability to handle high-dimensional data non-linear data, discriminate between classes, and generalize to different writing styles makes them suitable for this task. However, the effectiveness of SVMs depends on proper feature representation, parameter tuning, and the choice of kernel function. SVMs exhibit robustness to noise in the data. SVMs generalize well to different writing styles. The margin maximization objective helps SVMs find a decision boundary that minimizes overfitting, making them robust in handling variations in Devanagari handwriting. SVMs involve hyperparameters such as the regularization parameter (C) and the kernel parameters. Proper tuning of these parameters is essential for achieving optimal performance in Devanagari character recognition. Combining SVMs with other techniques or exploring ensemble approaches further enhances the recognition system's performance.

5) NEURAL NETWORK/MLP-BASED APPROACHES

An artificial neural network with several layers, comprising an input layer, one or more hidden layers, and an output layer,

TABLE 6. Neural network-based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	Data Set	Gap Analysis
R Bajaj, et al. [89]	Offline numeral recognition	2002	Kohonen nets for feature extraction	Meta-pi integration network	✗	Unknown	Limited benchmark data hindered larger dataset experimentation.
P Deshpande, et al. [159]		2006	Segment height, segment position, number of segments in one column and relationship between segments	Multilayer Perception neural networks	80%	A synthetic dataset of 150	Tested on limited data sets
S Arora, et al. [160]	Offline character recognition	2007	Structural properties like "shirorekha" and "spine" and intersection features	preliminary classification using structural properties	89.12 %	Synthetic data from 5000 samples	Challenges in detecting near-straight lines
S Arora, D Bhattacharjee, et al. [91]	Offline character recognition	2008	Intersection, shadow, chain code histogram, and straight-line fitting	weighted majority voting	92.80 %,	Table-1:dataset-[4]	Efficiency enhancement needed
S Kompalli, et al. [161]	Offline character recognition	2009	graph-based character segmentation, gradient, structural, and concavity feature	K-nearest neighbor stochastic finite state automaton for word recognition	96.97 %	ILT, EMILLE hindi text	A Hindi corpus has been used to design the language model. Corpora from other languages can be investigated.
S Arora, et al. [162,163]	Offline character recognition	2009, 2011	Chain code histogram, four side views, shadow-based	weighted majority voting	98.16 %	Own dataset of 1500 samples	Experiment with additional feature extraction methods for improved accuracy
P Goyal, S Diwakar and A Agrawal [164]	Offline character recognition	2010	Image segmentation	Kohonen Neural Network	90.26 % printed characters, 83.33 % handwritten characters	Unknown	Require optimizing neural network training
S. Arora, D. Bhattacharjee, et al. [165]	Offline character recognition	2010	Chain code, histogram, and moment-invariant features	✗	98.03 % for the top 5 results	Own dataset of 1500 samples	Explore alternative feature extraction methods
S Arora, L Malik, et al. [166]	Offline character recognition	2010	Directional gradient changes, intersections, spine type, and shirorekha type	✗	88.12 %	Own dataset of 1000 samples	A gap in achieving a higher recognition rate
S Arora et al. [101]	Offline character recognition	2010	Shadow, chain code histogram	Minimum edit distance	90.74 %	[4] and 2254 own-created samples	We need to work on more features and other classifiers
VP Agnihotri [167]	Offline character recognition	2012	Diagonal zone-based feature	✗	85.78 %	Own created 1000 samples	Need to use other classifiers like genetic algorithm optimization
RD Shelke and NP Patil [168]	Printed characters	2013	Moment invariants (MIs) and GLCM properties	✗	89.53 %	Own created 500 samples	Proposing integration of segmentation

TABLE 6. *(Continued.)* **Neural network-based approaches work done for handwritten devanagari characters.**

			like Contrast, Homogeneity, Entropy, Correlation, color domain, and histogram				and classification
N Sahu and NK Raman [169]	Offline character recognition	2013	Combination of stages of preprocessing, segmentation	✗	75.6%	Unknown	Continuous improvement through iterative testing for certain characters
A Dixit, A Navghane, and Y Dandawate [170]	Offline character recognition	2014	Wavelet transform and statistical parameters.	✗	70%	Own created 2000 samples.	Improvement in multistage feature extraction, exploring additional classification methods
N Singh [171]	Offline character recognition	2018	15 hidden layers and piecewise Histogram of Oriented Gradients (HOG)	✗	99.27 %	Unknown	Expanding to more characters and parsing approaches for entire documents

is called a Multi-Layer Perceptron (MLP). The advantages of the neural network classifier are its speed (very parallel capability), ease of training, and ability to partition the input feature space arbitrarily. MLPs are a feedforward neural network, meaning data goes from the input layer to the output layer in a single direction. MLP learns its parameters via a backpropagation process. The procedure of adjusting the weights and biases to reduce the loss is called backpropagation. Optimization algorithms like gradient descent and others entail calculating the gradient of the loss concerning the network's parameters and using this information to change the parameters.

Kohonen nets with a Meta-PI classifier show limited performance on benchmark data, with image encoding using histogram band analysis presenting a 3.25% average relative error but struggling with robust recognition. A Multilayer Perceptron (MLP) neural network utilizing segment height, position, number, and relationships achieves 80% accuracy but is tested on limited datasets. Structural properties and intersection features reach 89.12% accuracy but encounter challenges in detecting near-straight lines. Intersection, shadow, chain code histogram, and straight-line fitting with a weighted majority voting classifier achieve 92.80% accuracy but require efficiency enhancements. Graph-based character segmentation, gradient, structural, and concavity features with a K-nearest neighbor (KNN) stochastic finite state automaton classifier attain 96.97% accuracy. Shadow-based methods with a weighted majority voting classifier achieve 98.16% accuracy. Kohonen Neural Network-based image segmentation achieves 90.26% for printed characters and 83.33% for handwritten characters, demanding optimization in neural network training. Chain code, histogram, and moment invariants yield 98.03% accuracy for the top 5 results. Various directional features achieve accuracies ranging from 70% to 99.27%, with different methods demonstrating strengths and weaknesses in the preprocessing, segmentation, and classification stages.

Discussion: Multilayer Perceptrons (MLPs), an artificial neural network, offer a powerful and flexible approach for handwritten Devanagari character recognition. MLPs are capable of learning hierarchical representations of features from raw input data. In the context of handwritten Devanagari characters, this ability allows MLPs to automatically learn relevant patterns and representations from the pixel values of the character images. MLPs can capture complex, nonlinear relationships between input features and output classes. This flexibility is crucial in HDCR tasks, where the mapping between pixel values and character identity is not linear. The multiple layers and non-linear activation functions in MLPs allow them to capture intricate patterns in the data. MLPs involve hyperparameters such as the number of layers, number of neurons per layer, learning rate, and activation functions. Many researchers have experimented with properly tuning these hyperparameters to achieve optimal performance in Devanagari character recognition. MLPs are the most popularly used by researchers for HDCR tasks as they can be scaled up to handle large datasets and adapt to different writing styles and variations in the Devanagari character appearance. Recurrent Neural Networks (RNNs), a type of MLP, are particularly well-suited for capturing temporal dependencies and context in sequences, which is beneficial for recognizing Devanagari characters in words or sentences. MLPs might not inherently capture hierarchical structures present in Devanagari characters. CNNs, with their ability to automatically learn hierarchical features, are often more suitable.

6) FUZZY APPROACHES

Fuzzy logic comes to the fore, offering a robust and flexible solution. Fuzzy logic allows the representation of linguistic

variables closer to human reasoning. Linguistic variables such as ''loopiness,'' ''curviness,'' and ''angularity'' can be defined to describe the visual characteristics of handwritten characters. Fuzzy sets help in modeling the uncertainty and imprecision inherent in handwritten characters. Unlike rigid true/false classifications, fuzzy approaches utilize ''degrees of membership,'' allowing characters to partially belong to multiple classes based on features like shape, strokes, and proximity. This tolerance for variation accounts for messy handwriting and noisy scans while incorporating expert knowledge through fuzzy rules. The result is a powerful system that recognizes Devanagari characters with greater accuracy and adaptability, even amidst the inherent uncertainties of handwritten scripts.

Segmentation of touching characters uses a 98.87% accurate classifier. Structural and stroke features with a tree classifier have an accuracy of 86.4%. It enhances context recognition and rejection strategies. GLCM, color dominant, Affine moment invariant, and Histogram use an ANFIS classifier with an accuracy of 92.66%. Structural parameters have an accuracy of 96.95%, but they lack scalability. Fuzzy Directional Features, Discrete Wavelet Transform achieved 78.1% accuracy.

Discussion: Researchers' work reported on fuzzy approaches offer a flexible and human-interpretable framework for tackling the challenges of handwritten Devanagari character recognition. However, tuning fuzzy methods is challenging as these methods depend on effectively modeling linguistic variables, membership functions, and careful rule formulation.

7) DEEP LEARNING BASED APPROACHES

Deep learning is a subset of machine learning that leverages artificial neural networks inspired by the human brain's structure to learn and represent intricate patterns from data automatically. These neural networks consist of layers of interconnected nodes with adjustable weights and biases, enabling the model to process complex information hierarchically. The training process involves iterative adjustments to these parameters, guided by optimization algorithms, to minimize the difference between predicted and actual outcomes. Deep Neural Networks (DNNs) with multiple hidden layers allow for the automatic extraction of hierarchical features, while specialized architectures like Convolutional Neural Networks (CNNs) excel in image processing tasks, and Recurrent Neural Networks (RNNs) handle sequential data. In recent years, convolutional neural networks (CNNs) and long short-term memory (LSTM) networks have revolutionized character recognition. Researchers used deep learning architectures to produce outstanding results for handwritten Devanagari recognition challenges. Techniques for data augmentation and transfer learning also helped the performance.

BLSTM achieves a 20% lower word error rate and a 9% lower character error rate than a baseline model. However, it suffers from heavy degradation in accuracy and character confusion. Ten CNNs with additional techniques like Exponential Decay and inverse Scale Annealing achieves an accuracy of 98.19%. Deep Convolutional Neural Network (DCNN) with Dropout and data augmentation techniques achieves an accuracy of 98.47%. Deep Belief Network (DBN) with Stacked RBM achieves an accuracy of 83.44% with unsupervised learning and 91.81 after fine-tuning. Deep CNNs perform up to 98% accuracy. It achieves the

TABLE 7. Fuzzy approaches based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	DataSet	Gap Analysis
U Garain and BB Chaudhuri [172]	Printed character recognition	2002	segmentation of touching characters to select cut columns	✗	98.87 %	Documents collected from old books, etc.	It can also be applied to other scripts like Roman.
P Mukherji and PP Rege [173]	Offline character recognition	2009	Structural and stroke features, adaptive thinning, Average Compressed Direction Code (ACDC)	Tree classifier	86.4 %.	3 samples of 48 characters from 250 persons	Enhancement in context recognition and rejection strategies
GS Sable and SA Nirve [174]	Printed character recognition	2013	GLCM, color dominant, Affine moment invariant, and Histogram	ANFIS (Artificial neuro-fuzzy interference system)	92.66 %	Database generated using MATLAB	Highlight challenges of varied fonts and sizes in the Devanagari script
S Shelke and S Apte [175]	Offline character recognition	2015	Structural parameters	✗	96.95 %	more than 40,000 samples	Lacks the scalability
VL Lajish and SK Kopparapu [176]	Online character recognition	2010	Fuzzy Directional Features, Discrete Wavelet Transform	✗	78.1 %	Devanagari text paragraph by 10 different writers	Improve using Viterbi trace back and spatio-temporal information.

TABLE 8. Deep learning-based approaches work done for handwritten devanagari characters.

Author Name	Recognition	Year	Deep Learning Model	Additional techniques	Accuracy	DataSet	Gap Analysis
N Sankaran, et al [177]	Printed Devanagari text	2012	Bidirectional Long Short-Term Memory (BLSTM)	✗	20% lower word error rate and 9% lower character error rate	Manually typed dataset: training - 90K, validation-74K, testing-67K words	Difficulties in heavy degradation and character confusion
Mehrotra K, et al [51]	Online handwritten Devanagari characters	2013	Ten Convolutional Neural Networks	Exponential Decay and Inverse Scale Annealing	98.19 %	41359 extracted characters	Extend classes and explore word and sentence recognition.
S Acharya, et al [178]	New dataset Devanagari Handwritten Character Dataset	2015	Deep Convolutional Neural Network	Dropout and dataset increment techniques	98.47 %	Table-1:dataset-[1]	Expanding the dataset to include vowels, derived consonants, and special characters
S Prabhanjan, et al [179]	Offline Handwritten Character	2017	Deep Belief Network (DBN)	Stacked Restricted Boltzmann Machines (RBM) for feature learning	83.44 % with unsupervised learning and 91.81 % after fine-tuning	Table-1:dataset-[4],Table-1:dataset-[5](2012) and their own dataset	Work on other deep-learning techniques
M Jangid, et al [180]	Handwritten Devanagari character	2018	Deep convolutional neural networks	RMSProp. Layer-wise training	98%	Table-1:dataset-[4] and Table-1:dataset-[9]	Additional optimization techniques and Work on diverse datasets
S Ram, et al. [181]	Offline character recognition	2018	Deep convolutional neural networks	optimizing hyperparameters for network performance	96.9%	Table-1:dataset-[1]	The best performance was achieved by an 8-layer CNN using the ReLU activation function and dropouts.
Ghosh R, et al. [182]	Online handwritten word recognition	2019	Long-Short Term Memory (LSTM)	Bidirectional LSTM (BLSTM)	99.50 %	Created 750 words of data	Extension to sentence and paragraph recognition.
N Aneja, et al [183]	Offline character recognition	2019	Deep Convolutional Neural Network	Transfer learning with pre-trained models (AlexNet, DenseNet, Vgg, Inception)	AlexNet performs the fastest, with 98% accuracy in 6.6 minutes over three epochs.	Table-1:dataset- [1]	Optimizing model architectures for smaller datasets.
SP Deore et al. [184]	Offline character recognition	2020	CNN	Two-stage VGG16	96.55 %	5800 characters of 12 vowels, 36 consonants, and 10 numerals	Explore different Deep CNN for compound characters
DS Prashanth, et al [185]	Offline Handwritten	2022	Convolutional Neural Networks	Modified Lenet CNN and Alexnet	MLCNN - 99% and 94%	Table-1:dataset- [3]	Other Deep-learning techni

TABLE 8. *(Continued.)* **Deep learning-based approaches work done for handwritten devanagari characters.**

	Character		(CNN)	CNN	accuracy ACNN - 99% and 98% accuracy		questcan be experimented with.
A Moudgil, et al [186]	Devanagari manuscript characters	2023	CNN model	CapsNet	94.6%	400 ancient manuscripts: 240 - Pehowa and 160 - Haridwar.	Scalability and robustness for enhanced performance
B Yadav, et al [187]	Offline handwritten Devanagari characters	2023	CNN-based model, HDevCharNet	Batch Normalization (BN) and dropout layers.	99.17 %	Table-1:dataset- [1]	Need work to handle vowels and modifiers
A Jindal, et al [188]	Offline Handwritten Character	2023	LSTM	Bi-LSTM	96.97 %	Self-generated dataset	We need to work on other variants of RNN
Akhter SS et al. [189]	Offline printed Devanagari script	2023	Hyperparameter tuning, Taguchi's method	RNN	Improved WER by 1.78%, CER by 1.8%, CTC loss by 5.7 value	IIIT-ILST dataset, Synthetic 10,000 pictures, newspaper articles	Works on 32 × 128-sized word images, work on another dataset for testing

best performance with techniques like ReLU activation and dropout. LSTMs with Bidirectional extension (BLSTM) achieve high accuracy (99.5%) in sentence and paragraph recognition tasks. Transfer Learning with pre-trained models like AlexNet can achieve high accuracy (98%). The two-stage VGG16 model achieves 96.55% accuracy in recognizing compound characters. Modified LeNet CNN and AlexNet CNN achieve high accuracy (about 99%). CapsNet achieves 94.6% accuracy on tasks requiring high scalability and robustness. A CNN-based model with Batch Normalization (BN) achieves high accuracy for tasks involving vowels and modifiers. LSTM with Bi-LSTM is 96.97% accurate.

Deep learning approaches have been tried to recognize handwritten Farsi digits. A CBWME model [196] was developed, integrating CNN with bagging weighted majority ensemble learning, using VGG16, ResNet18, and Xception as base classifiers. The CBWME model achieved the highest recognition accuracy of 97.65% on the HODA dataset. In financial Farsi documents, traditional classifiers (KNN, ANN, SVM) and advanced methods (CNN, auto-encoder) have been implemented [197]. CNN achieved the highest accuracy (99.45%), surpassing SVM's best accuracy of 99.3% with improved performance and calculation time.

Discussion: When applied to handwritten Devanagari character recognition, deep learning techniques use neural network architectures to learn hierarchical features from raw data automatically. Deep learning models are currently the best-performing models for handwritten Devanagari character recognition. CNNs learn to recognize stroke patterns, curves, and other distinctive features, providing robust performance across different writing styles. RNNs, with their ability to model sequential dependencies, are suitable for capturing temporal aspects in handwritten character strokes. Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) cells within RNNs address the vanishing gradient problem and improve the learning of temporal dependencies. Hybrid architectures that combine the strengths of both CNNs and RNNs have successfully recognized spatial and sequential features in handwritten characters. For example, a system might use a CNN for feature extraction followed by an RNN to model sequential dependencies.

Transfer learning, leveraging pre-trained models on large datasets, has been beneficial in domains where labeled data is scarce. Pre-trained models on general character datasets or other scripts can be fine-tuned for Devanagari character recognition. This approach is useful when limited labeled data is available for Devanagari characters.

Attention mechanisms have been successfully applied to improve the performance of deep learning models for sequence recognition tasks. They allow the model to focus on relevant parts of the input sequence, which is beneficial in recognizing complex handwritten characters.

Deep learning approaches often require large labeled datasets for practical training, which is a limitation in the case of Devanagari character recognition due to the script's complexity and variations. Interpretability of deep learning models is also challenging, making it difficult to understand how and why a specific recognition decision is made. However, addressing challenges related to the availability of labeled data and model interpretability remains necessary for further advancements in this field. A comprehensive evaluation should go beyond accuracy and consider efficiency and robustness metrics such as Execution Time, Complexity, Delay Time, Confusion Matrix, True Positive Rate (TPR), False Positive Rate (FPR), ROC and AUC-ROC, and Cohen's

TABLE 9. Other approaches.

Technique Name	Author Name	Recognition	Year	Feature Extraction method	Additional classifier	Accuracy	DataSet	Gap Analysis
Genetic Algorithm	V Saraf and DS Rao [194]	Offline Handwritten Character	2013	✗	✗	98.78 %,	Unknown	Limited work on Indian script
Random Forest	K Johnson, et al. [195]	Devanagari numerals	2017	Zonal Histogram of Angles	Random Forest	92.57 %	Table-1:dataset-[4]	Digits 2, 3, and 5 showed frequent misclassifications
Soft computing techniques	A Chaudhuri, et al [196]	Offline Handwritten Character	2017	Fuzzy Hough transform	RFMLP, FSVM, FRSVM, FMRF	RFMLP - 99% FSVM and FRSVM - 97% FMRF - 99.8 %	270 samples of each of the 111 Devnagari characters	Asymmetries, slants, and uneven pixel concentration in characters.
Extra Trees classifier	SM Pande and BK Jha [197]	Offline Handwritten Character	2021	✗	Random Forest classifiers	Extra Trees classifiers-78.19 %, Random Forest classifier - 76.82 %	Offline database: 43 thousand images (32 × 32 pixels)	Hybrid approach to enhance model accuracy.
K-nn classifier	MMN Mayekar, et al. [198]	Offline Handwritten Character	2017	✗	GPU, Parallel processing with CUDA	Accuracy is good with a higher value of k	Training: 61,200 and Testing: 10,800 images	Trade-off between speed and accuracy
Fisher linear discriminant model$	M Jangid, et al [199]	Offline Handwritten Character	2018	Discrimination of similar character	✗	Critical regions were detected well.	Table-1:dataset-[4]	Appropriate classifiers combined improve accuracy.

Kappa. While models like BLSTM achieve up to 99.5% accuracy, they may suffer from character confusion and high complexity. Metrics like AUC-ROC and Cohen's Kappa provide a better understanding of model performance beyond simple accuracy. In real-world applications, execution time and delay are critical, particularly for complex models like CNNs and transfer learning approaches like AlexNet. Thus, evaluating these models across broader criteria ensures their practical efficiency in real-time deployment while maintaining robustness and high recognition accuracy.

8) OTHER COMBINED APPROACHES

Apart from the techniques mentioned above for Devanagari characters and numeral recognition, few researchers worked on tree-based classifiers, genetics algorithms, and soft computing techniques. The reported work is summarized in Table 9.

Genetic Algorithm has 98.78% accuracy but with limited work on Indian scripts. Random Forest with Zonal Histogram of Angles has an accuracy of 92.57%. It frequently misclassified digits 2, 3, and 5. Soft Computing Techniques include Fuzzy Hough Transform, RFMLP, FSVM, FRSVM, and FMRF classifiers, achieving an accuracy of 97% to 99.8% depending on the classifier used. Extra Trees Classifier uses Random Forest classifiers and achieves an accuracy of 78.19% from 76.82%.

Discussion: Diverse machine learning algorithms have shown promise in character recognition research with distinctive strengths and limitations. The Genetic Algorithm, while impressively accurate overall, requires refinement for improved performance on Indian scripts. The Random Forest algorithm, utilizing the Zonal Histogram of Angles, demonstrates robust performance but faces challenges in accurately classifying specific digits. Soft Computing Techniques, including the Fuzzy Hough Transform, RFMLP, FSVM, FRSVM, and FMRF classifiers, prove versatile in achieving consistently high accuracies. The Extra Trees Classifier, an extension of Random Forest, notably enhances accuracy. The k-NN classifier, leveraging GPU and parallel processing, achieves commendable accuracy with a trade-off between speed and precision. Additionally, the Fisher Linear Discriminant Model enhances accuracy when combined judi-

TABLE 10. Performance-based comparison of different techniques.

Technique	Highest Accuracy	Description
Structural and Statistical approach	98.6 %	Dataset was very small
Template-matching based approach	95%	dataset was unknown, 97.29 % for online characters
Hidden Markov Model (HMM) based approach	93.82 %	Data was 350 words only
Support Vector Machine (SVM) based approach	99.1 %	7515 character images, other SVM work on different features also performed well compared to other traditional techniques
Neural Network/MLP-based approach	99.27 %	dataset was unknown
Fuzzy approach	98.87 %	Printed characters, unknown size
Deep Learning based approach	99.50 %	LSTM based model on 750 words

ciously with appropriate classifiers, especially in identifying critical regions. This array of techniques emphasizes the multifaceted nature of character recognition, highlighting the importance of a nuanced approach considering both global and specific contextual factors.

9) PERFORMANCE COMPARISON OF CHARACTER RECOGNITION TECHNIQUES

This comparison highlights the accuracy and performance of various character recognition techniques, ranging from traditional methods like Template-matching to advanced approaches such as Deep Learning. The results showcase the effectiveness of modern machine learning models in achieving higher accuracy rates.

The comparison shows that modern approaches like Deep Learning (LSTM) and Neural Networks outperform traditional techniques such as Template-matching and Hidden Markov Models in character recognition tasks. The Deep Learning approach achieved the highest accuracy (99.5%), indicating its superiority, especially with larger or more complex datasets. Support Vector Machines and Neural Networks also demonstrated strong performance, suggesting that advanced machine learning methods are highly effective for this application. Traditional methods like Structural and Statistical approaches still provide good results but are limited by dataset size and complexity.

10) IMPLEMENTATION TOOLS

From 1980 to 2024, the literature on handwritten Devanagari character recognition shows how software environments have evolved to reflect advances in OCR technology. Because of its intuitive toolboxes, MATLAB was often used for early prototyping and simple image processing jobs. Java and Tesseract OCR provided a strong foundation for printed text recognition in the 2000s, while OpenCV became a potent toolkit for image processing and feature extraction. Python's ascent in the middle of the 2010s, together with libraries like TensorFlow, Keras, and PyTorch, transformed the field and enabled sophisticated deep-learning models like CNNs and RNNs. Concurrently, C/C++ was used to create specialized OCR algorithms because of its processing efficiency. While cloud-based systems like Google Colab and Amazon SageMaker made it easier to train large-scale models, tools like WEKA and R gained popularity for classical machine learning approaches. Libraries for transfer learning and data augmentation have also become more significant in improving model performance, especially when dealing with small labeled datasets. This development demonstrates the shift in Devanagari character recognition towards increasingly complex machine learning-based methods.

11) ROLE OF DEPTH AND WIDTH IN DEVANAGARI CHARACTER RECOGNITION MODELS

In the context of Devanagari character recognition, the depth and width parameters play a crucial role in determining the model's capacity and performance. Depth and width are generally measured for neural network- or deep learning-based models. Depth refers to the number of layers in the neural network, which impacts the model's ability to capture complex hierarchical patterns and subtle features within Devanagari characters. A deeper network can learn more intricate representations, capturing finer details of character variations and strokes, which is essential for distinguishing similar-looking characters. However, excessive depth can lead to issues like vanishing gradients or overfitting, especially when data is limited.

On the other hand, width denotes the number of neurons or units in each layer, influencing the model's ability to capture a broader range of features within each layer. Adequate width allows the model to learn distinct aspects of each character, enhancing its robustness in recognizing diverse handwriting styles. Therefore, balancing depth and width is essential; an optimal configuration enables the model to effectively learn the unique features of handwritten Devanagari characters while maintaining computational efficiency and preventing overfitting, ultimately boosting recognition accuracy.

In the context of deep learning models like CNN, VGG16, ResNet18, and LSTM for Devanagari character recognition, depth and width parameters distinctly influence each model's performance and suitability for this task. Convolutional Neural Networks (CNNs), such as VGG16, rely heavily on depth, where each additional layer enhances the model's capacity to capture detailed spatial hierarchies in characters. VGG16, with its deeper architecture and uniform layer width, excels in extracting intricate patterns essential for distinguishing visually similar Devanagari characters, though it can be

computationally intensive. ResNet18 introduces residual connections to address the challenges of deep networks, like vanishing gradients, allowing it to capture complex features at substantial depths without the risk of degrading performance. These residual connections are precious in character recognition, as they help retain fine-grained information over deep layers. LSTMs, however, utilize width in terms of hidden units to capture sequential dependencies, making them practical for processing character sequences but less commonly applied to static character images alone. Depth in LSTM networks allows for more prosperous temporal feature extraction, which can be valuable in tasks like text recognition across multiple characters. Balancing depth and width across these architectures is critical: CNNs and ResNet can benefit from added depth for better spatial feature learning, while LSTMs rely on width for robust sequence representation, ultimately enhancing accuracy in recognizing Devanagari characters.

III. FUTURE DIRECTIONS

The field of Devanagari character recognition is dynamic and continues to evolve with advancements in technology and research. End-to-end recognition systems integrate segmentation, feature extraction, and classification. Such systems can jointly optimize the entire recognition pipeline, potentially improving overall performance.

- Architectures like transformers-based models or novel convolutional neural network (CNN) structures may offer improvements in capturing intricate patterns and hierarchical features within Devanagari characters.
- Hybrid architectures that combine the strengths of both CNNs and RNNs have successfully recognized spatial and sequential features in handwritten characters. For example, a system might use a CNN for feature extraction followed by an RNN to model sequential dependencies.
- Integration of attention mechanisms within recognition systems is another direction of research. Attention mechanisms can help the model focus on relevant parts of characters, which is especially beneficial for handling variations in writing styles.
- Exploration of unsupervised learning techniques for Devanagari character recognition needs attention as unsupervised learning can help in scenarios where labeled data is limited, allowing models to discover patterns and features without explicit supervision.
- Further utilization of transfer learning and pre-training on large datasets is also an open area of research. Pre-training models on diverse datasets can help capture generic features, which can be fine-tuned for Devanagari character recognition, especially in scenarios with limited labeled data.
- Synthetic data generated by GANs can diversify training datasets, potentially improving model robustness to variations in writing styles and data quality. Systems can be researched that adapt to different writing styles, varying image qualities, and other real-world challenges may perform better in diverse scenarios. Multimodal Approaches to Combine visual information with contextual cues can provide additional context for character recognition, especially in scenarios where visual information alone might be ambiguous.
- As technology advances, addressing the challenges specific to Devanagari character recognition and exploring innovative solutions will likely lead to improved performance and broader applicability in real-world scenarios.

IV. CONCLUSION

Handwritten character recognition is the most challenging area of pattern recognition, and it has multiple applications. It can contribute immensely to advancing an automation process and improve the interface between humans and machines in many applications. Intensive research has been done on Handwritten Character Recognition, and many articles have been published on this topic. However, no explicit review work has yet been reported on Handwritten Devanagari character recognition. This paper extensively reviews handwritten character recognition work done on Devanagari characters. All the techniques of preprocessing, segmentation, and feature selections specifically used for Devanagari characters have been presented on which researchers have worked. A timeline of all the work is also offered. Recognition of Devanagari characters or numerals using various techniques such as statistical, structural, machine learning, and deep learning has been extensively reviewed. We believe that our extensive literature review will strongly encourage activities of Handwritten Devanagari document processing.

REFERENCES

[1] J. Mantas, "An overview of character recognition methodologies," *Pattern Recognit.*, vol. 19, no. 6, pp. 425–430, Jan. 1986.
[2] S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of OCR research and development," *Proc. IEEE*, vol. 80, no. 7, pp. 1029–1058, Jul. 1992.
[3] S. Mori, K. Yamamoto, and M. Yasuda, "Research on machine recognition of hand-printed characters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 4, pp. 386–405, Jul. 1984.
[4] G. Nagy, "At the frontiers of OCR," *Proc. IEEE*, vol. 80, no. 7, pp. 1093–1100, Jul. 1992.
[5] R. Plamondon and S. N. Srihari, "On-line and off-line handwritten recognition: A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, Jan. 2000.
[6] S. N. Srihari, J. J. Hull, and S. C. Shapiro, "Character recognition," in *Encyclopedia of Artificial Intelligence*. Hoboken, NJ, USA: Wiley, 1992, pp. 138–150.
[7] G. Nagy and S. Seth, "Modern optical character recognition," in *The Froehlich/Kent Encyclopedia Telecommunication*, F. E. Froehlich and A. Kent., New York, NY, USA: Marcel Dekker, 1996, pp. 473–531.
[8] *Kurzweil Reading Machine for the Blind*, R. C. Kurzwei, Cambridge, MA, USA, 1990.
[9] A. Amin, "Off-line Arabic character-recognition: The state of the art," *Pattern Recognit.*, vol. 31, pp. 517–530, Jul. 1998.
[10] F. El-Khaly and M. A. Sid-Ahmed, "Machine recognition of optically captured machine printed Arabic text," *Pattern Recognit.*, vol. 23, no. 11, pp. 1207–1214, Jan. 1990.
[11] T. S. El-Sheikh and R. M. Guindi, "Computer recognition of Arabic cursive scripts," *Pattern Recognit.*, vol. 21, no. 4, pp. 293–302, Jan. 1988.

[12] G. Nagy, "Chinese character recognition—A twenty-five years retrospective," in *Proc. 9th Int. Conf. Pattern Recognit.*, 1988, pp. 109–114.
[13] L. O. Gorman and R. Kasturi, *Document Image Analysis*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1995.
[14] J. Rocha and T. Pavlidis, "Character recognition without segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 9, pp. 903–909, Sep. 1995.
[15] W. Stallings, "Approaches to Chinese character recognition," *Pattern Recognit.*, vol. 8, no. 2, pp. 87–98, Apr. 1976.
[16] Ø. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition—A survey," *Pattern Recognit.*, vol. 29, no. 4, pp. 641–662, Apr. 1996.
[17] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, Jan. 2000.
[18] J. A. Pittman, "Handwriting recognition: Tablet PC text input," *Computer*, vol. 40, no. 9, pp. 49–54, Sep. 2007.
[19] H. Nishida, "An approach to integration of off-line and on-line recognition of handwriting," *Pattern Recognit. Lett.*, vol. 16, no. 11, pp. 1213–1219, Nov. 1995.
[20] G. Boccignone, A. Chianese, L. P. Cordella, and A. Marcelli, "Recovering dynamic information from static handwriting," *Pattern Recognit.*, vol. 26, no. 3, pp. 409–418, Mar. 1993.
[21] D. S. Doermann and A. Rosenfeld, "Recovery of temporal information from static images of handwriting," *Int. J. Comput. Vis.*, vol. 15, nos. 1–2, pp. 143–164, Jun. 1995.
[22] R. Plamondon and C. M. Privitera, "The segmentation of cursive handwriting: An approach based on off-line recovery of the motor-temporal information," *IEEE Trans. Image Process.*, vol. 8, no. 1, pp. 80–91, Jan. 1999.
[23] F. Nouboud and R. Plamondon, "On-line recognition of handprinted characters: Survey and beta tests," *Pattern Recognit.*, vol. 23, no. 9, pp. 1031–1044, Jan. 1990.
[24] R. Plamondon, T. D. Lopresti, R. B. Schomaker, and R. Srihari, "Online handwriting recognition," in *Encyclopedia of Electrical and Electronics Engineering*, vol. 15. Hoboken, NJ, USA: Wiley, 1979, pp. 13–146.
[25] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of art in online handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 8, pp. 787–808, Aug. 1990.
[26] T. Wakahara, H. Murase, and K. Odaka, "On-line handwriting recognition," *Proc. IEEE*, vol. 80, no. 7, pp. 1181–1194, Jul. 1992.
[27] M. Yadav, R. K. Purwar, and M. Mittal, "Handwritten Hindi character recognition: A review," *IET Image Process.*, vol. 12, no. 11, pp. 1919–1933, Nov. 2018.
[28] R. Singh, R. K. Mishra, S. S. Bedi, S. Kumar, and A. K. Shukla, "A literature review on handwritten character recognition based on artificial neural network," *Int. J. Comput. Sci. Eng.*, vol. 6, no. 11, pp. 753–758, Nov. 2018.
[29] M. Kumar, M. K. Jindal, and R. K. Sharma, "Review on OCR for handwritten Indian scripts character recognition," in *Proc. Int. Conf. Digit. Image Process. Inf. Technol.*, 2011, pp. 268–276.
[30] A. K. Bathla, S. K. Gupta, and M. K. Jindal, "Challenges in recognition of devanagari scripts due to segmentation of handwritten text," in *Proc. 3rd Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, Mar. 2016, pp. 2711–2715.
[31] P. Hirugade, N. Suryavanshi, R. Bhagwat, S. Rajput, and R. Phadke, "A survey on optical character recognition for handwritten devanagari script using deep learning," in *Proc. Int. Conf. Innov. Comput. Commun.*, 2022, pp. 1–4.
[32] M. Agrawal, B. Chauhan, and T. Agrawal, "Machine learning algorithms for handwritten devanagari character recognition: A systematic review," *J. Sci. Technol.*, vol. 7, no. 1, pp. 1–16, Jul. 2023.
[33] A. Sharma and B. N. Mithun, "Deep learning character recognition of handwritten devanagari script: A complete survey," in *Proc. IEEE Int. Conf. Contemp. Comput. Commun. (InC4)*, Apr. 2023, pp. 1–6.
[34] M. Bhatnagar, "Review for handwritten devanagari character recognition using ML algorithms," *Int. J. Adv. Res. Sci., Commun. Technol.*, vol. 2020, pp. 71–86, Jun. 2020.
[35] S. Khare and J. Singh, "Handwritten devanagari character recognition system: A review," *Int. J. Comput. Appl.*, vol. 121, no. 9, pp. 10–14, Jul. 2015.
[36] V. J. Dongre and V. H. Mankar, "Development of comprehensive devnagari numeral and character database for offline handwritten character recognition," in *Proc. Appl. Comput. Intell. Soft Comput.*, 2012, pp. 1–5.
[37] P. A. Sharma, "Review on devanagari character recognition," *IJRAR-Int. J. Res. Anal. Rev.*, vol. 5, no. 3, pp. 473–478, 2018.
[38] V. Vijay, M. U. Kharat, and S. V. Gumaste, "Study of different features and classification techniques for recognition of handwritten devanagari text," *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 1055–1059, Nov. 2018.
[39] P. Guruprasad, "Variety threshold treatment for handwritten devanagari word," in *Proc. 3rd Nat. Conf. Comput. Appl.*, Jun. 2020, pp. 1–8.
[40] S. Arora, S. Jahirabadkar, and A. Kulkarni, "GPU approach for handwritten devanagari document binarization," in *Smart Innovations in Communication and Computational Sciences* (Advances in Intelligent Systems and Computing), vol. 670. Singapore: Springer, 2019.
[41] S. Prabhanjan and R. Dinesh, "Handwritten devanagari numeral recognition by fusion of classifiers," *Int. J. Signal Process., Image Process. Pattern Recognit.*, vol. 8, no. 7, pp. 41–50, Jul. 2015.
[42] P. M. Yawalkar and M. U. Kharat, "Automatic handwritten character recognition of devanagari language: A hybrid training algorithm for neural network," *Evol. Intell.*, vol. 15, no. 2, pp. 1499–1516, Jun. 2022.
[43] T. A. Jundale and R. S. Hegadi, "Research survey on skew detection of Devanagari script," *Int. J. Comput. Appl.*, vol. 975, p. 8887, May 2015.
[44] T. A. Jundale and R. S. Hegadi, "Skew detection and correction of devanagari script using Hough transform," *Proc. Comput. Sci.*, vol. 45, pp. 305–311, Jul. 2015.
[45] T. A. Jundale and R. S. Hegadi, "Skew detection and correction of Devanagari script using interval halving method," in *Recent Trends in Image Processing and Pattern Recognition RTIP2R Communications in Computer and Information Science*, vol. 709. Singapore: Springer, 2017.
[46] T. A. Jundale and R. S. Hegadi, "Skew detection of devanagari script using pixels of axes-parallel rectangle and linear regression," in *Proc. Int. Conf. Energy Syst. Appl.*, Oct. 2015, pp. 480–484.
[47] R. Thapliya, M. Tsuchiya, and T. Kamiya, "Joint transform correlator applied to recognition of devanagari script," *Opt. Rev.*, vol. 3, no. 6, pp. 397–399, Nov. 1996.
[48] A. K. Bathla and S. K. Gupta, "Character segmentation and skew correction for handwritten devanagari scripts: A friends technique," *Asian J. Eng. Appl. Technol.*, vol. 8, no. 1, pp. 50–54, Feb. 2019.
[49] D. S. Guru, M. Suhil, M. Ravikumar, and S. Manjunath, "Small eigenvalue based skew estimation of handwritten Devanagari words," in *Proc. Int. Conf. Mining Intell. Knowl. Explor.*, 2015, pp. 216–225.
[50] R. Kapoor, D. Bagai, and T. S. Kamal, "Skew angle detection of a cursive handwritten Devanagari script character image," *J. Indian Inst. Sci.*, vol. 82, no. 3, pp. 75–161, May 2002.
[51] K. Mehrotra, S. Jetley, A. Deshmukh, and S. Belhe, "Unconstrained handwritten devanagari character recognition using convolutional neural networks," in *Proc. 4th Int. Workshop Multilingual OCR*, Aug. 2013, pp. 1–5.
[52] S. D. Chowdhury, U. Bhattacharya, and S. K. Parui, "Online handwriting recognition using Levenshtein distance metric," in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 79–83.
[53] S. Singh, N. K. Garg, and M. Kumar, "Feature extraction and classification techniques for handwritten devanagari text recognition: A survey," *Multimedia Tools Appl.*, vol. 82, no. 1, pp. 747–775, Jan. 2023.
[54] P. Deshpande, L. Malik, and S. Arora, "Character recognition with histogram band analysis of encoded string and neural network," in *Proc. 4th WSEAS Int. Conf. Inf. Secur., Commun. Comput.*, Jan. 2005, pp. 354–359.
[55] D. Khanduja, N. Nain, and S. Panwar, "A hybrid feature extraction algorithm for devanagari script," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 15, no. 1, pp. 1–10, Jan. 2016.
[56] P. Yawalkar and M. U. Kharat, "Effective thinning algorithm for recognition of handwritten Devanagari compound characters using neural network," *Int. J. Appl. Eng. Res.*, vol. 13, no. 12, pp. 50–10539, 2018.
[57] G. Goyal and M. Dutta, "Design of pixel neighborhood based offline handwritten thinning framework for Devanagari numeral script using Elman neural network," *Int. J. Comput. Sci. Inf. Secur.*, vol. 14, no. 7, pp. 1–7, 2016.
[58] K. Manpreet and B. A. Kumar, "Segmentation of characters of devanagari script documents," *World Wide J. Multidisc. Res. Dev.*, vol. 3, no. 11, pp. 253–257, 2017.
[59] C. Amit, "A review of various character segmentation techniques for cursive handwritten words recognition," *Int. J. Inf. Comput. Technol.*, vol. 4, no. 6, pp. 559–564, 2014.
[60] T. Saba, G. Sulong, and A. Rehman, "Document image analysis: Issues, comparison of methods and remaining problems," *Artif. Intell. Rev.*, vol. 35, pp. 101–118, Feb. 2011.

[61] M. Holt, M. Beglou, and S. Datta, "Slant-independent letter segmentation for off-line cursive script recognition," in *Proc. Pixels Features III*, S. Impedovo and J. C. Simon, Eds., 1992, pp. 41–42.

[62] P. S. Lakshmi, M. Hanmandlu, and A. Swaroop, "Segmentation of cursive handwritten words using hypergraph," in *Proc. IEEE Region 10 Conf. (TENCON)*, Aug. 2006, pp. 1–4.

[63] A. Dawoud, "Iterative cross section sequence graph for handwritten character segmentation," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2150–2154, Aug. 2007.

[64] H. Lee and B. Verma, "A novel multiple experts and fusion based segmentation algorithm for cursive handwriting recognition," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 2994–2999.

[65] L. Rabiner, "A tutorial on hidden Markov models and selected speech recognition applications," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[66] P. R. Cavalin, A. de Souza Britto, F. Bortolozzi, R. Sabourin, and L. E. S. Oliveira, "An implicit segmentation-based method for recognition of handwritten strings of characters," in *Proc. ACM Symp. Appl. Comput.*, Apr. 2006, pp. 836–840.

[67] V. J. Dongre and V. H. Mankar, "Devanagari document segmentation using histogram approach," *Int. J. Comput. Sci., Eng. Inf. Technol.*, vol. 1, no. 3, pp. 1–12, Aug. 2011.

[68] V. J. Dongre, V. H. Mankar, and G. Suganya, "A review of research on devnagari character recognition," *Int. J. Comput. Appl.*, vol. 12, no. 2, pp. 8–15, Dec. 2010.

[69] K. Y. Wong, R. G. Casey, and F. M. Wahl, "Document analysis system," *IBM J. Res. Develop.*, vol. 26, no. 6, pp. 647–656, Nov. 1982.

[70] L. Likforman-Sulem, A. Zahour, and B. Taconet, "Text line segmentation of historical documents: A survey," *Int. J. Document Anal. Recognit. (IJDAR)*, vol. 9, nos. 2–4, pp. 123–138, Apr. 2007.

[71] V. Bansal and R. M. Sinha, "Integrating knowledge sources in the Devanagari text recognition system," *IEEE Trans. Syst., Man, Cybern. A, Syst. Hum.*, vol. 30, no. 4, pp. 5–500, Jul. 2000.

[72] R. Sarkar, B. Sen, N. Das, and S. Basu, "Handwritten devanagari script segmentation: A non-linear fuzzy. Approach," in *Proc. IEEE Conf. AI Tools Eng.*, 2008, pp. 1–14.

[73] D. V. Sharma and G. S. Lehal, "An iterative algorithm for segmentation of isolated handwritten words in gurmukhi script," in *Proc. IEEE Int. Conf. ON Pattern Recognit.*, vol. 2, 2006, pp. 1022–1025.

[74] B. Thakral and M. Kumar, "Devanagari handwritten text segmentation for overlapping and conjunct characters—A proficient technique," in *Proc. 3rd Int. Conf. Rel., Infocom. Technol. Optim.*, Oct. 2014, pp. 1–4.

[75] X. Peng, H. Cao, S. Setlur, V. Govindaraju, and P. Natarajan, "Multilingual OCR research and applications: An overview," in *Proc. 4th Int. Workshop Multilingual OCR*, 2013, pp. 1–8.

[76] A. Jindal and R. Ghosh, "Word and character segmentation in ancient handwritten documents in devanagari and maithili scripts using horizontal zoning," *Expert Syst. Appl.*, vol. 225, Sep. 2023, Art. no. 120127.

[77] A. Jindal and R. Ghosh, "Text line segmentation in Indian ancient handwritten documents using faster R-CNN," *Multimedia Tools Appl.*, vol. 82, no. 7, pp. 22–10703, 2023.

[78] D. Trier, A. K. Jain, and T. Taxt, "Feature extraction method for character recognition: A survey," *Pattern Recognit.*, vol. 29, no. 4, pp. 641–662, 1996.

[79] A. Nafiz and F. T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *IEEE Trans. Syst., Man, Cybern., C, Appl. Rev.*, vol. 31, no. 2, pp. 216–233, Feb. 2000.

[80] G. G. Rajput and S. M. Mali, "Fourier descriptor based isolated Marathi handwritten numeral recognition," *Int. J. Comput. Appl.*, vol. 3, no. 4, pp. 9–13, Jun. 2010.

[81] U. Bhattacharya and B. B. Chaudhuri, "A majority voting scheme for multiresolution recognition of handprinted numerals," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, vol. 1, 2003, pp. 16–20.

[82] M. Mercimek, K. Gulez, and T. V. Mumcu, "Real object recognition using moment invariants," *Sadhana*, vol. 30, no. 6, pp. 765–775, Dec. 2005.

[83] R. J. Ramteke and S. C. Mehrotra, "Recognition of handwritten Devnagari numerals," *Int. J. Comput. Process. Oriental Lang.*, vol. 1, no. 2, pp. 1–9, Jan. 2014.

[84] K. V. Kale, D. P. Deshmukh, S. V. Chavan, M. M. Kazi, and Y. S. Rode, "Zernike moment feature extraction for handwritten Devanagari (Marathi) compound character recognition," *IJARAI Int. J. Adv. Res. Artif. Intell.*, vol. 3, no. 1, pp. 459–466, Jan. 2014.

[85] S. Chaudhury, G. Sethi, A. Vyas, and G. Harit, "Devising interactive access techniques for Indian language document images," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, vol. 1, 2003, pp. 885–889.

[86] R. M. K. Sinha, "On partitioning a dictionary for visual text recognition," *Pattern Recognit.*, vol. 23, no. 5, pp. 497–500, Jan. 1990.

[87] B. V. Dhandra and M. Hangarge, "Global and local features based handwritten text words and numerals script identification," in *Proc. Int. Conf. Comput. Intell. Multimedia Appl. (ICCIMA)*, Dec. 2007, pp. 471–475.

[88] T. K. Bhowmik, S. K. Parui, and U. Roy, "Discriminative HMM training with GA for handwritten word recognition," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[89] R. Bajaj, L. Dey, and S. Chaudhury, "Devnagari numeral recognition by combining decision of multiple connectionist classifiers," *Sadhana*, vol. 27, no. 1, pp. 59–72, Feb. 2002.

[90] F. Kimura and M. Shridhar, "Handwritten numeral recognition based on multiple algorithms," *Pattern Recognit.*, vol. 24, no. 10, pp. 969–983, 1991.

[91] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, and M. Kundu, "Combining multiple feature extraction techniques for handwritten devnagari character recognition," in *Proc. IEEE Region 10 3rd Int. Conf. Ind. Inf. Syst.*, Dec. 2008, pp. 1–6.

[92] N. Sharma, U. Pal, F. Kimura, and S. Pal, "Recognition of off-line handwritten Devanagari characters using quadratic classifier," in *Proc. 5th Indian Conf. Comput. Vis., Graph. Image Process. (ICVGIP)*, Dec. 2006, pp. 805–816.

[93] V. Mane and L. Ragha, "Handwritten character recognition using elastic matching and PCA," in *Proc. Int. Conf. Adv. Comput., Commun. Control*, Jan. 2009, pp. 410–415.

[94] C. Kruengkrai, V. Sornlertlamvanich, and H. Isahara, "Language, script, and encoding identification with string kernel classifiers," in *Proc. Conf. Knowl., Inf. Creativity Support Syst.* Thailand: Thai Computational Linguistics Laboratory, 2018, pp. 1–8.

[95] M. S. A. Bhopi and M. M. P. Singh, "Feature extraction techniques for Marathi character classification using neural networks models," *Int. J. Future Revolution Comput. Sci. Commun. Eng.*, vol. 4, no. 6, pp. 70–76, 2011.

[96] P. S. Bodkhe and P. E. Ajmire, "Analytical study of statistical features extraction of characters for verification of CAPTCHA in devanagari script," in *Proc. IJCRT*, 2020, vol. 8, no. 2, pp. 1–11.

[97] S. V. Rajashekararadhya and P. V. Ranjan, "A novel zone based feature extraction algorithm for handwritten numeral recognition of four Indian scripts," *Digit. Technol., J.*, vol. 2, pp. 41–51, Jun. 2009.

[98] P. Singh and S. Budhiraja, "Feature extraction and classification techniques in OCR system for handwritten gurmukhi script survey," *Int. J. Eng. Res. Appl.*, vol. 1, no. 4, pp. 1734–1739, Aug. 2012.

[99] K. M. Kim, J. J. Park, Y. G. Song, I. Kim, and C. Y. Suen, "Recognition of handwritten numerals using a combined classifier with hybrid features," in *Proc. Joint (IAPR) Int. Workshops Stat. Techn. Pattern Recognit. (SPR) Struct. Syntactic Pattern Recognit. (SSPR)*, vol. 3138, Jan. 2004, pp. 992–1000.

[100] N. Arica and F. T. Yarman-Vural, "Optical character recognition for cursive handwriting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 6, pp. 801–813, Jun. 2002.

[101] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, and M. Kundu, "Recognition of non-compound handwritten devnagari characters using a combination of MLP and minimum edit distance," *Int. J. Comput. Sci. Secur.*, vol. 4, no. 7, p. 106, Jan. 2010.

[102] P. Chavan, S. Sankpal, A. Sonawane, S. Shaikh, and A. Raut, "Handwritten devnagari optical character recognition," *Int. J. Innov. Res. Comput. Sci. Technol.*, vol. 2, no. 2, pp. 1–6, Mar. 2014.

[103] B. K. Verma, "Handwritten Hindi character recognition using multilayer perceptron and radial basis function neural networks," in *Proc. Int. Conf. Neural Netw. (ICNN95)*, vol. 4, 1995, pp. 2111–2115.

[104] P. S. Deshpande, L. Malik, and S. Arora, "Handwritten devnagari character recognition using connected segments and minimum edit distance," in *Proc. IEEE Region 10 Conf. (TENCON)*, Taipei, Taiwan, Oct. 2007, pp. 1–4.

[105] P. S. Deshpande, L. Malik, and S. Arora, "Characterizing hand written devanagari characters using evolved regular expressions," in *Proc. IEEE Region 10 Conf. (TENCON)*, Nov. 2006, pp. 1–4.

[106] P. S. Deshpande, L. Malik, and S. Arora, "Fine classification & recognition of hand written devnagari characters with regular expressions & minimum edit distance method," *J. Comput.*, vol. 3, no. 5, pp. 7–11, May 2008.

[107] S. Bag and G. Harit, "Topographic feature extraction for Bengali and Hindi character images," *Int. J. Signal Image Process.*, vol. 2, no. 2, pp. 181–196, 2011.
[108] I. K. Sethi and B. Chatterjee, "Machine recognition of hand-printed devnagri numerals," *IETE J. Res.*, vol. 22, no. 8, pp. 532–535, Aug. 1976.
[109] R. M. K. Sinha and H. N. Mahabala, "Machine recognition of Devanagari script," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 8, pp. 435–441, Aug. 1979.
[110] R. M. K. Sinha, "Role of context in Devanagari script recognition," *IETE J. Res.*, vol. 33, no. 3, pp. 86–91, May 1987.
[111] R. M. K. Sinha and V. Bansal, "On Devanagari document processing," in *Proc. IEEE Int. Conf. Syst., Man Cybern. Intell. Syst. 21st Century*, vol. 2, Jul. 1995, pp. 1621–1626.
[112] K. Keeni, H. Shimodaira, T. Nishino, and Y. Tan, "Recognition of devanagari characters using neural networks," *IEICE Trans. Inf. Syst.*, vol. 79, no. 5, pp. 8–523, May 1996.
[113] B. B. Chaudhuri and U. Pal, "An OCR system to read two Indian language scripts: Bangla and devnagari (Hindi)," in *Proc. 4th Int. Conf. Document Anal. Recognit.*, vol. 2, 1997, pp. 1011–1015.
[114] U. Pal and B. B. Chaudhuri, "Automatic separation of machine-printed and hand-written text lines," in *Proc. 5th Int. Conf. Document Anal. Recognit. (ICDAR)*, 1999, pp. 645–648.
[115] V. Bansal and R. M. K. Sinha, "On how to describe shapes of devanagari characters and use them for recognition," in *Proc. 5th Int. Conf. Document Anal. Recognit. (ICDAR)*, 1999, pp. 410–413.
[116] S. D. Connell, R. M. K. Sinha, and A. K. Jain, "Recognition of unconstrained online devanagari characters," in *Proc. 15th Int. Conf. Pattern Recognit. ICPR-*, vol. 2, 2000, pp. 368–371.
[117] K. Jayanthi, A. Suzuki, H. Kanai, Y. Kawazoe, M. Kimura, and K. Kido, "Devanagari character recognition using structure analysis," in *Proc. 4th IEEE Region 10 Int. Conf. (TENCON)*, Nov. 1989, pp. 363–366.
[118] A. Baldominos, Y. Saez, and P. Isasi, "A survey of handwritten character recognition with MNIST and EMNIST," *Appl. Sci.*, vol. 9, no. 15, p. 3169, Aug. 2019.
[119] N. Sethi, A. Dev, and P. Bansal, "A novel neural machine translation approach for low-resource sanskrit-Hindi language pair," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 2023, pp. 1–19, Apr. 2023.
[120] A. Ranade and M. Ranade, "Devanagari pen-written character recognition," in *Proc. 9th Int. Conf. Adv. Comput. Commun. (ADCOM)*, Dec. 2001, pp. 1–7.
[121] U. Pal and B. B. Chaudhuri, "Machine-printed and hand-written text line identification," *Pattern Recognit. Lett.*, vol. 22, no. 3, pp. 431–441, Mar. 2001.
[122] V. Bansal and R. M. K. Sinha, "A complete OCR for printed Hindi text in devanagari script," in *Proc. 6th Int. Conf. Document Anal. Recognit.*, 2001, pp. 800–804.
[123] U. Pal and N. Tripathy, "Recognition of Indian multi-oriented and curved text," in *Proc. 8th Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Aug. 2005, pp. 141–145.
[124] U. Pal, N. Sharma, T. Wakabayashi, and F. Kimura, "Off-line handwritten character recognition of devanagari script," in *Proc. 9th Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Sep. 2007, pp. 496–500.
[125] R. J. Ramteke, "Invariant moments based feature extraction for handwritten devanagari vowels recognition," *Int. J. Comput. Appl.*, vol. 1, no. 18, pp. 1–5, Feb. 2010.
[126] M. Jangid, "Devanagari isolated character recognition by using statistical features," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 2, pp. 2400–2407, Jun. 2011.
[127] L. Malik, "A graph based approach for handwritten devanagari word recognition," in *Proc. 5th Int. Conf. Emerg. Trends Eng. Technol.*, Nov. 2012, pp. 309–313.
[128] A. Deshmukh, R. Meshram, S. Kendre, and K. Shah, "Handwritten devanagari character recognition," *Int. J. Eng. Res.*, vol. 3, no. 4, pp. 1–26, Apr. 2014.
[129] S. Gaikwad, S. Nalbalwar, and A. Nandgaonkar, "Recognition of offline handwritten devanagari characters using new mask-based approach, histogram of oriented gradients and AdaBoost," *Multimedia Tools Appl.*, vol. 82, no. 28, pp. 43883–43902, Apr. 2023.
[130] K. C. Santosh, C. Nattee, and B. Lamiroy, "Relative positioning of stroke-based clustering: A new approach to online handwritten devanagari character recognition," *Int. J. Image Graph.*, vol. 12, no. 2, Apr. 2012, Art. no. 1250016.
[131] U. Garain, B. B. Chaudhuri, and T. T. Pal, "Online handwritten Indian script recognition: A human motor function based framework," in *Proc. Object Recognit. Supported User Interact. Service Robots*, vol. 3, Aug. 2002, pp. 164–167.
[132] H. Ma and D. Doermann, "Adaptive Hindi OCR using generalized Hausdorff image comparison," *ACM Trans. Asian Lang. Inf. Process.*, vol. 2, no. 3, pp. 193–218, Sep. 2003.
[133] V. K. Nandury, "Stroke based recognition of online Devanagari handwritten characters," Doctoral dissertation, Indian Inst. Sci. Bengaluru, Bangalore, India, 2005.
[134] P. S. Deshpande, L. Malik, and S. Arora, "Recognition of hand written devnagari characters with percentage component regular expression matching and classification tree," in *Proc. 10th IEEE Region Conf. (TENCON)*, Oct. 2007, pp. 1–4.
[135] L. Malik and P. S. Deshpande, "Recognition of handwritten Devanagari script," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 24, no. 5, pp. 22–809, Aug. 2010.
[136] P. Natarajan, E. MacRostie, and M. Decerbo, "The BBN byblos Hindi OCR system," *Document Recognit. Retr. XII*, vol. 5676, pp. 173–180, Jan. 2009.
[137] U. Bhattacharya, S. K. Parui, B. Shaw, and K. Bhattacharya, "Neural combination of ANN and HMM for handwritten devanagari numeral recognition," in *Proc. 10th Int. Workshop Frontiers Handwriting Recognit.*, Oct. 2006, pp. 1–7.
[138] S. K. Parui and B. Shaw, "Offline handwritten devanagari word recognition: An hmm-based approach," in *Proc. 2nd Int. Conf. Pattern Recognit. Mach. Intell. (PReMI)*, Dec. 2007, pp. 528–535.
[139] B. Shaw, S. K. Parui, and M. Shridhar, "Offline handwritten devanagari word recognition: A holistic approach based on directional chain code feature and HMM," in *Proc. Int. Conf. Inf. Technol.*, Dec. 2008, pp. 203–208.
[140] B. Shaw and S. K. Parui, "A two stage recognition scheme for offline handwritten devanagari words," *Mach. Interpretation Patterns, Image Anal. Data Mining*, vol. 2010, pp. 145–165, Jun. 2010.
[141] M. A. Shaikh and M. R. Dagade, "Offline recognition of handwritten devanagari words using hidden Markov model," *Int. J. Innov. Res. Sci. Technol.*, vol. 1, no. 11, pp. 1–6, 2015.
[142] R. Ghosh and P. P. Roy, "Comparison of zone-features for online Bengali and devanagari word recognition using HMM," in *Proc. 15th Int. Conf. Frontiers Handwriting Recognit. (ICFHR)*, Oct. 2016, pp. 435–440.
[143] R. Ghosh and P. P. Roy, "Study of zone-based feature for online handwritten signature recognition and verification in devanagari script," in *Proc. Int. Conf. Comput. Vis. Image Process.*, Dec. 2016, pp. 523–530.
[144] C. V. Jawahar, M. P. Kumar, and S. R. Kiran, "A bilingual OCR for Hindi-telugu documents and its applications," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, vol. 1, 2003, pp. 408–412.
[145] U. Pal, S. Chanda, T. Wakabayashi, and F. Kimura, "Accuracy improvement of devnagari character recognition combining SVM and MQDF," in *Proc. 11th Int. Conf. Frontiers Handwrit. Recognit*, Aug. 2008, pp. 367–372.
[146] S. A. D. Bhattacharjee, M. Nasipuri, L. Malik, M. Kundu, and D. K. Basu, "Performance comparison of SVM and ANN for handwritten Devanagari character recognition," *IJCSI Int. J. Comput. Sci. Issues*, vol. 7, no. 3, p. 18, May 2010.
[147] B. Singh, A. Mittal, M. A. Ansari, and D. Ghosh, "Handwritten devanagari word recognition: A curvelet transform based approach," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 4, pp. 1658–1665, Apr. 2011.
[148] M. Jangid, R. Dhir, and R. Rani, "Novel approach: Recognition of devanagari handwritten numerals," *Int. J. Elect., Electron. Comput. Sci.*, vol. 1, no. 2, pp. 6–41, 2011.
[149] M. Jangid, R. Dhir, R. Rani, and K. Singh, "SVM classifier for recognizing handwritten devanagari numerals," in *Proc. Int. Conf. Image Inf. Process.*, Nov. 2011, pp. 1–5.
[150] M. Jangid and S. Srivastava, "Gradient local auto-correlation for handwritten devanagari character recognition," in *Proc. Int. Conf. High Perform. Comput. Appl. (ICHPCA)*, Dec. 2014, pp. 1–5.
[151] A. Gaur and S. Yadav, "Handwritten Hindi character recognition using k-means clustering and SVM," in *Proc. 4th Int. Symp. Emerg. Trends Technol. Libraries Inf. Services*, Jan. 2015, pp. 65–70.
[152] R. Ghosh and P. P. Roy, "Study of two zone-based features for online Bengali and devanagari character recognition," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 401–405.
[153] N. Tripathy, T. Chakraborti, M. Nasipuri, and U. Pal, "A scale and rotation invariant scheme for multi-oriented character recognition," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 4041–4046.

[154] P. K. Singh, S. Das, R. Sarkar, and M. Nasipuri, "Recognition of offline handwriten devanagari numerals using regional weighted run length features," in *Proc. Int. Conf. Comput., Electr. Commun. Eng. (ICCECE)*, Dec. 2016, pp. 1–6.

[155] S. R. Narang, M. K. Jindal, and P. Sharma, "Devanagari ancient character recognition using HOG and DCT features," in *Proc. 5th Int. Conf. Parallel, Distrib. Grid Comput. (PDGC)*, Dec. 2018, pp. 215–220.

[156] S. Deore and A. Pravin, "Histogram of oriented gradients based off-line handwritten devanagari characters recognition using SVM, K-NN and NN classifiers," *Revue d'Intell. Artificielle*, vol. 33, no. 6, pp. 441–446, Dec. 2019.

[157] S. R. Narang, M. K. Jindal, and M. Kumar, "Devanagari ancient character recognition using DCT features with adaptive boosting and bootstrap aggregating," *Soft Comput.*, vol. 23, no. 24, pp. 13603–13614, Dec. 2019.

[158] A. Kumar, S. Bhatia, M. R. Khosravi, A. Mashat, and P. Agarwal, "Semantic and context understanding for sentiment analysis in Hindi handwritten character recognition using a multiresolution technique," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 23, no. 1, pp. 1–22, Jan. 2024.

[159] P. S. Deshpande, L. Malik, and S. Arora, "Character recognition using a relationship between connected segments and neural network," *WSEAS Trans. Comput.*, vol. 5, no. 1, pp. 34–229, Jan. 2006.

[160] S. Arora, D. Bhatcharjee, M. Nasipuri, and L. Malik, "A two stage classification approach for handwritten devnagari characters," in *Proc. Int. Conf. Comput. Intell. Multimedia Appl. (ICCIMA)*, Dec. 2007, pp. 399–403.

[161] S. Kompalli, S. Setlur, and V. Govindaraju, "Devanagari OCR using a recognition-driven segmentation framework and stochastic language models," *Int. J. Document Anal. Recognit. (IJDAR)*, vol. 12, pp. 38–123, Jul. 2009.

[162] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, M. Kundu, and L. Malik, "Study of different features on handwritten devnagari character," in *Proc. 2nd Int. Conf. Emerg. Trends Eng. Technol.*, 2009, pp. 929–933.

[163] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, and M. Kundu, "Complementary features combined in a MLP-based system to recognize handwritten devanagari character," *J. Inf. Hiding Multimedia Signal Process.*, vol. 2, no. 1, pp. 7–71, Jan. 2011.

[164] P. Goyal, S. Diwakar, and A. Agrawal, "Devanagari character recognition towards natural human-computer interaction," *India HCI Interact. Design Int. Develop.*, vol. 2010, pp. 1–5, Mar. 2010.

[165] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, and M. Kundu, "Application of statistical features in handwritten devnagari character recognition," 2010, *arXiv:1006.5911*.

[166] S. Arora, L. Malik, D. Bhattacharjee, and M. Nasipuri, "A novel approach for handwritten devnagari character recognition," 2010, *arXiv:1006.5924*.

[167] V. P. Agnihotri, "Offline handwritten devanagari script recognition," *Int. J. Inf. Technol. Comput. Sci.*, vol. 4, no. 8, pp. 37–42, Jul. 2012.

[168] R. D. Shelke and N. P. Patil, "A neural network approach to printed devanagari character recognition," *Int. J. Comput. Appl., pp.*, vol. 975, p. 8887, 2013.

[169] N. Sahu and N. K. Raman, "An efficient handwritten devnagari character recognition system using neural network," in *Proc. Int. Mutli-Conf. Autom., Comput., Commun., Control Compressed Sens. (iMac4s)*, Mar. 2013, pp. 173–177.

[170] A. Dixit, A. Navghane, and Y. Dandawate, "Handwritten devanagari character recognition using wavelet based feature extraction and classification scheme," in *Proc. Annu. IEEE India Conf. (INDICON)*, Dec. 2014, pp. 1–4.

[171] N. Singh, "An efficient approach for handwritten devanagari character recognition based on artificial neural network," in *Proc. 5th Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Feb. 2018, pp. 894–897.

[172] U. Garain and B. B. Chaudhuri, "Segmentation of touching characters in printed devnagari and Bangla scripts using fuzzy multifactorial analysis," *IEEE Trans. Syst., Man Cybern., C, Appl. Rev.*, vol. 32, no. 4, pp. 449–459, Nov. 2002.

[173] P. Mukherji and P. P. Rege, "Shape feature and fuzzy logic based offline devnagari handwritten optical character recognition," *J. Pattern Recognit. Res.*, vol. 5, no. 1, pp. 52–68, 2010.

[174] G. S. Sable and S. A. Nirve, "Optimization of optical character recognition for printed devanagari text using ANFIS techniques," *Int. J. Comput. Netw.*, vol. 3, no. 5, p. 3, 2013.

[175] S. Shelke and S. Apte, "A fuzzy based classification scheme for unconstrained handwritten devanagari character recognition," in *Proc. Int. Conf. Commun., Inf. Comput. Technol. (ICCICT)*, Jan. 2015, pp. 1–6.

[176] V. L. Lajish and S. K. Kopparapu, "Fuzzy directional features for unconstrained on-line devanagari handwriting recognition," in *Proc. Nat. Conf. Commun. (NCC)*, Jan. 2010, pp. 1–5.

[177] N. Sankaran and C. V. Jawahar, "Recognition of printed devanagari text using BLSTM neural network," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, Nov. 2012, pp. 322–325.

[178] S. Acharya, A. K. Pant, and P. K. Gyawali, "Deep learning is based on large-scale handwritten devanagari character recognition," in *Proc. 9th Int. Conf. Softw., Knowl., Inf. Manage. Appl. (SKIMA)*, Dec. 2015, pp. 1–6.

[179] S. Prabhanjan and R. Dinesh, "Deep learning approach for devanagari script recognition," *Int. J. Image Graph.*, vol. 17, no. 3, Jul. 2017, Art. no. 1750016.

[180] M. Jangid and S. Srivastava, "Handwritten devanagari character recognition using layer-wise training of deep convolutional neural networks and adaptive gradient methods," *J. Imag.*, vol. 4, no. 2, p. 41, Feb. 2018.

[181] S. Ram, S. Gupta, and B. Agarwal, "Devanagari character recognition model using deep convolutional neural network," *J. Statist. Manage. Syst.*, vol. 21, no. 4, pp. 593–599, Jul. 2018.

[182] R. Ghosh, C. Vamshi, and P. Kumar, "RNN-based online handwritten word recognition in Devanagari and Bengali scripts using horizontal zoning," *Pattern Recognit.*, vol. 92, pp. 203–218, Aug. 2019.

[183] N. Aneja and S. Aneja, "Transfer learning using CNN for handwritten devanagari character recognition," in *Proc. 1st Int. Conf. Adv. Inf. Technol. (ICAIT)*, Jul. 2019, pp. 293–296.

[184] S. P. Deore and A. Pravin, "Devanagari handwritten character recognition using fine-tuned deep convolutional neural network on trivial dataset," *Sādhanā*, vol. 45, no. 1, pp. 1–13, Dec. 2020.

[185] D. S. Prashanth, R. V. K. Mehta, K. Ramana, and V. Bhaskar, "Handwritten devanagari character recognition using modified lenet and alexnet convolution neural networks," *Wireless Pers. Commun.*, vol. 122, no. 1, pp. 349–378, Jan. 2022.

[186] A. Moudgil, S. Singh, V. Gautam, S. Rani, and S. H. Shah, "Handwritten devanagari manuscript characters recognition using capsnet," *Int. J. Cognit. Comput. Eng.*, vol. 4, pp. 47–54, Jun. 2023.

[187] B. Yadav, A. Indian, and G. Meena, "HDevChaRNet: A deep learning-based model for recognizing offline handwritten devanagari characters," *J. Auto. Intell.*, vol. 6, no. 2, p. 679, Aug. 2023.

[188] A. Jindal and R. Ghosh, "A hybrid deep learning model to recognize handwritten characters in ancient documents in devanagari and maithili scripts," *Multimedia Tools Appl.*, vol. 83, no. 3, pp. 8389–8412, Jun. 2023.

[189] S. S. M. N. Akhter and P. P. Rege, "Hyper parameter optimization of CRNN for printed devanagari script recognition using Taguchi's method," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 22, no. 4, pp. 1–20, Mar. 2023.

[190] V. Saraf and D. S. Rao, "Devanagari script character recognition uses genetic algorithms to achieve better efficiency," *Int. J. Soft Comput. Eng.*, vol. 2013, pp. 2231–2307, Apr. 2013.

[191] K. Johnson, K. Gourav, Gaurav, D. Rudrapal, and S. Debnath, "OCR for devanagari numerals using zonal histogram of angle," *J. Statist. Manage. Syst.*, vol. 20, no. 4, pp. 519–534, Jul. 2017.

[192] A. Chaudhuri, K. Mandaviya, P. Badelia, and S. K. Ghosh, "Optical character recognition systems for Hindi language," *Opt. Character Recognit. Syst. Different Lang. Soft Comput.*, vol. 2017, pp. 193–216, Dec. 2017.

[193] S. M. Pande and B. K. Jha, "Character recognition system for devanagari script using machine learning approach," in *Proc. 5th Int. Conf. Comput. Methodologies Commun. (ICCMC)*, Apr. 2021, pp. 899–903.

[194] Miss. M. N. Mayekar and Mrs. S. Kuwelkar, "Implementation of machine learning algorithm for character recognition on GPU," in *Proc. Int. Conf. Comput. Methodologies Commun. (ICCMC)*, Jul. 2017, pp. 470–474.

[195] M. Jangid and S. Srivastava, "Handwritten devanagari similar character recognition by Fisher linear discriminant and pairwise classification," *Int. J. Image Graph.*, vol. 18, no. 4, Oct. 2018, Art. no. 1850022.

[196] Y. A. Nanehkaran, J. Chen, S. Salimi, and D. Zhang, "A pragmatic convolutional bagging ensemble learning for recognition of Farsi handwritten digits," *J. Supercomput.*, vol. 77, no. 11, pp. 13474–13493, Nov. 2021.

[197] Y. A. Nanehkaran, D. Zhang, S. Salimi, J. Chen, Y. Tian, and N. Al-Nabhan, "Analysis and comparison of machine learning classifiers and deep neural networks techniques for recognition of Farsi handwritten digits," *J. Supercomput.*, vol. 77, no. 4, pp. 3193–3222, Apr. 2021.

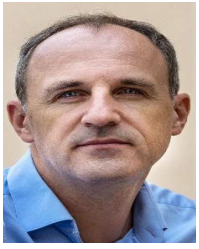
SANDHYA ARORA received the B.E. degree in computer engineering from the University of Rajasthan, India, the M.Tech. degree in computer science engineering from Banasthali Vidyapith, Rajasthan, India, and the Ph.D. degree in computer science and engineering from Jadavpur University, Kolkata, in 2012. She is a Full Professor with the Department of Computer Engineering, Cummins College of Engineering for Women, Pune. She has teaching experience of more than 26 years. She has more than 60 research publications. Also, she has authored seven books under University Press, India, and CRC Press, USA. She is a Life Member of ISTE, CSI, and ACM.

LATESH MALIK received the B.E. degree in computer engineering from the University of Rajasthan, India, the M.Tech. degree in computer science and engineering from Banasthali Vidyapith, Rajasthan, India, and the Ph.D. degree in computer science and engineering from the Visvesyaraya National Institute of Technology, in 2010. She is an Associate Professor and the Head of the Department of Computer Science and Engineering, Government College of Engineering, Nagpur. She is also the Chairperson of the Board of Studies, Computer Engineering related branch board, R.T.M. Nagpur University, from 2022 to 2027. She has teaching experience of 25 years. She has more than 160 papers published in international journals and conferences. She has guided more than 30 PG projects and eight students completed Ph.D. under her guidance. She is the author of seven books under University Press, India, and CRC Press, USA. She is a Life Member of ISTE, CSI, and ACM. She was a Gold Medalist in B.E. and M.Tech. studies. She was a recipient of two RPS and one MODROBs by AICTE.

SONAKSHI GOYAL received the B.Tech. degree in computer engineering, in 2023. She is an Alumnus of MKSSS's Cummins College of Engineering for Women, Pune, Maharashtra. She is with J. P. Morgan Hyderabad.

DEBOTOSH BHATTACHARJEE (Senior Member, IEEE) is a Full Professor with the Department of Computer Science and Engineering, Jadavpur University. He has 17 years of post-Ph.D. experience. He has been a Visiting Professor and a Postdoctoral Researcher with various universities abroad. He has been granted sponsored projects by the Government of India funding agencies for a total amount of around INR 30 Million. He has authored or co-authored more than 132 journals, 142 conference proceedings publications, and 31 book chapters in biometrics and medical image processing. Two U.S. patents have been granted on his works. His research interests include the applications of machine learning techniques for face recognition, gait analysis, hand geometry recognition, and diagnostic image analysis. He is a fellow of the West Bengal Academy of Science Technology; and a Life Member of the Indian Society for Technical Education (ISTE), New Delhi, and the Indian Unit for Pattern Recognition and Artificial Intelligence (IUPRAI). He is on the editorial board of several journals.

MITA NASIPURI (Life Senior Member, IEEE) received the B.E.Tel.E., M.E.Tel.E., and Ph.D. (Eng.) degrees from Jadavpur University, Calcutta, India, in 1979, 1981, and 1990, respectively. She is currently a Professor with the Computer Science and Engineering Department, Jadavpur University. She has 30 research publications in international/national journals and international/national conferences. Her research interests include computer architecture, image processing, multimedia systems, and biomedical signal processing. She is a fellow of the Institution of Engineers, India.



ONDREJ KREJCAR received the Ph.D. degree in technical cybernetics from the Technical University of Ostrava, Czech Republic, in 2008. He has been a Rector and a Full Professor with Skoda Auto University, since 2024. He is also affiliated with the Faculty of Informatics and Management, University of Hradec Kralove (UHK), Czech Republic. He is also the Director of the Center for Basic and Applied Research, UHK. From 2020 to 2024, he was the Vice-Rector of science and creative activities with UHK. From 2016 to 2020, he was the Vice-Dean of science and research with the Faculty of Informatics and Management, UHK. At UHK, he is a guarantee of the doctoral study program in applied informatics, where he is focusing on lecturing on smart approaches to the development of information systems and applications in ubiquitous computing environments. His H-index is 33 at Web of Science, where more than 200 IF journal articles are indexed in the JCR-index (H-index is 38 at SCOPUS with more than 6400 citations). His research interests include technical cybernetics, ubiquitous computing, control systems, smart sensors, wireless technology, biomedicine, image segmentation and recognition, biometrics, biotelemetric system architecture (portable device architecture and wireless biosensors), and the development of applications for mobile/remote devices with the use of remote or embedded biomedical sensors. In 2018, he was the 14th Top Peer Reviewer in Multidisciplinary in the World, according to Publons, and a Top Reviewer in the Global Peer Review Awards 2019 by Publons. He was a Vice-Leader and a Management Committee Member at WG4 for Project COST CA17136, from 2018 to 2024. He has also been a Management Committee Member Substitute at project COST CA16226, from 2017 to 2023. Since 2019, he has been the Chairperson of the Program Committee of the KAPPA Program, Technological Agency of the Czech Republic, as a regulator of the EEA/Norwegian Financial Mechanism in the Czech Republic (2019–2024). Since 2020, he has also been the Chairperson of Panel 1 (Computer, Physical and Chemical Sciences) of the ZETA Program, Technological Agency of the Czech Republic. From 2014 to 2019, he was the Deputy Chairperson of the Panel 7 (Processing Industry, Robotics, and Electrical Engineering) of the Epsilon Program, Technological Agency of the Czech Republic. He is on the editorial board of several journals indexed in Q1/Q2 at JCR.