A Reduced Feature-Set OCR System to Recognize Handwritten Tamil Characters using SURF Local Descriptor

Ashlin Deepa R N[1], S.Sankara Narayanan[2], Adithya Padthe[3], Manjula Ramannavar[4]
Department of Computer Science and Engineering,
Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, India[1]
Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur[2]
University of Cumberlands, USA[3]
Department of Computer Science and Engineering, KLS Gogte Institute of Technology, Affiliated to Visvesvaraya Technological University Udyambag, Belagavi, Karnataka[4]

***Abstract*—High dimensionality in variable-length feature sets of real datasets negatively impacts the classification accuracy of traditional classifiers. Convolutional Neural Networks (CNNs) with convolution filters have been widely used for handling the classification of high-dimensional image datasets. However, these models require massive amounts of high-dimensional training data, posing a challenge for many image-processing applications. In contrast, traditional feature detectors and descriptors, with a minor trade-off in precision, have shown success in various computer vision tasks. This paper introduces the Nearest Angles (NA) classifier tailored for a handwritten character recognition system, employing Speeded-Up Robust Features (SURF) as local descriptors. These descriptors make local decisions, while global decisions on the test image are accomplished through a ranking-based classification approach. Image similarity scores generated from the SURF descriptors are ranked to make local decisions, and these ranks are then used by the NA classifier to produce a global class similarity score. The proposed method achieves recognition rates of 96.4% for Tamil, 96.5% for Devanagari, and 97 % for Telugu handwritten character datasets. Although the proposed approach shows slightly lower accuracy compared to CNN-based models, it significantly reduces the computational complexity and the number of parameters required for the classification tasks. As a result, the proposed method offers a computationally efficient alternative to deep learning models, lowering the computational time multiple times without a substantial loss in accuracy.**

Keywords—Image processing; feature extraction; Convolutional Neural Networks; SURF; handwritten character recognition; optical character recognition system

I. Introduction

Human-machine interaction is increasingly becoming a cornerstone in various applications that span daily life. Among these applications, Optical Character Recognition (OCR) stands out as a key technology that facilitates the conversion of different types of handwritten or printed characters into machine-encoded text, generally in ASCII or UNICODE formats [1]. This technology finds various applications ranging from document digitization to automated data entry. For economically deprived and rural populations in India, an effective handwritten OCR system in native languages can significantly enhance computer usability and information accessibility, thereby bridging the digital divide [2].

Despite advances in OCR technologies, there are still significant challenges that affect the performance of OCR systems for handwritten texts, especially in Indian languages [3] [4]. These challenges include inconsistencies in writing styles, structural similarity between symbols, and noise in input images [5] [6]. In addition, existing solutions such as Convolutional Neural Networks (CNNs) require massive computational resources and extensive high-dimensional data for training, making them less feasible for resource-constrained environments [7] [8] [9]. Moreover, CNN-based solutions might not be effective in handling variable-length feature sets or high-dimensional data without a large corpus of training samples [10] [11] [12]. Speeded-Up Robust Features (SURF) descriptors are a type of feature descriptor that is commonly used in image processing and OCR [13]. However, they have been shown to be less effective for handwritten text, due to the structural variations and noise that can be present in input images. This can lead to inaccurate feature extraction, which can in turn degrade the performance of the OCR system. Singular value decomposition (SVD) is a technique for reducing the dimensionality of high-dimensional data [14]. It is often used in OCR to improve the performance of feature extraction and classification. However, SVD can be computationally expensive, especially for large-scale applications. This can limit the practical applicability of SVD-based OCR systems.

The primary aim of this research is to tackle the challenges associated with the OCR system for handwritten characters in Indian languages by introducing a computationally efficient and yet accurate classifier. Specifically, this study introduces the Nearest Angles (NA) classifier, designed for a handwritten character recognition system. This classifier operates in tandem with SURF as local descriptors. Unlike CNN-based solutions that need to process high-dimensional data as a whole, the NA classifier breaks down the problem into more manageable parts by making local decisions based on the SURF descriptors.

The SURF descriptors are employed to extract features from the handwritten characters. These descriptors capture local features from images, making them less sensitive to writing inconsistencies and noise. Upon feature extraction, the NA classifier comes into play. The classifier uses a ranking-based classification approach that relies on generating image similarity scores from the SURF descriptors. These scores are then ranked for making local decisions. Based on the ranks, the NA classifier generates a global class similarity score that is used for classifying the test data. The overall method leverages local decision-making to reduce the computational burden and achieves classification accuracy rates of 91.0% for Tamil, 94.7% for Devanagari, and 88% for Telugu handwritten character datasets.

1) Introduction of nearest angles (NA) classifier: Developed a new classifier specifically tailored for handwritten character recognition systems, which is computationally more efficient than existing deep learning approaches.

2) Integration with SURF descriptors: Demonstrated the efficacy of using SURF as local descriptors in combination with the NA classifier, thus offering a balanced trade-off between computational complexity and classification accuracy.

3) Ranking-based classification approach: Introduced a unique ranking-based classification method that generates a global class similarity score based on local decisions, simplifying the classification task without a significant loss in accuracy.

4) Multilingual support: Validated the approach across multiple Indian languages including Tamil, Devanagari and Telugu, thus showing its versatility and wide applicability.

5) Resource efficiency: Demonstrated that the proposed method significantly reduces the computational time and the number of parameters required for classification tasks, making it suitable for resource-constrained environments.

These contributions collectively highlight the novel aspects of this research, showcasing its importance in the fields of OCR and machine learning, especially in the context of Indian languages and resource-constrained environments.

II. Literature Review

First, the Indian scripts lack a sound HCR system due to their complex nature. Most image processing applications use deep-learning-based models such as CNN and produce promising results [15]. The algorithms for deep learning have substantially improved, functioning exceptionally in a variety of fields with robust safety features [16], [17], [18]. In the context of supervised learning, CNN needs a lot of data to enhance its ability to learn and generalization. Else, it leads to overfitting and data shortage [19]. CNNs have emerged as the method of choice for computer vision tasks involving classification, clustering, and other image-processing applications. But here, general-purpose object recognition tasks frequently appear to produce low-level characteristics [20]. Many studies are made to evaluate the complexity of CNN models. In a recent study on the time complexity of eight deep-learning models, the number of convolution layers, filters, pooling layers, and fully connected layers are varied and tested. The authors found that these layers directly affect the system's performance [21]. Speeded-Up Robust Features (SURF) [22] provides a patented local feature detector and descriptor. Handwritten character images vary in shape and size, and each image has a distinct number of SURF descriptions derived from it. That means the feature vector of each character image is variable in length. Since most classifiers are intended to use fixed-length strings, a variable-length vector of features poses a challenge. On the part of handling variable length feature vectors, Singular Value Decomposition (SVD) is used along with SVM [23]. An approach in which feature vectors are the chromosomes of GA where two variable length strings are handled by padding one of them with {1,0,#} and the classifier was defined as VGA-classifier [24]. In the previous approach called 'Modified-GA', feature vectors are variable length in nature and are handled by appending '00' at the end [25]. However, these methods amplify the complexity in terms of storing and processing the information.

To incorporate a deep-learning-based classification approach, Surf-CNN was proposed, in which local features are extracted by SURF, and CNN is used for classification [20]. In a study, the presence and location of crack information in concrete structures are determined using SURF features and CNN [26]. When SIF and SURF descriptions are incorporated, the deep-learning-based methods can outperform traditional VGG16 and MobileNetV2 [27]. The number of parameters used in CNN is huge which leads to complexity in the classification systems which can be avoided if the accuracy can be slightly compromised.

Simple algorithms such as Nearest Neighbor (NN) can address a variety of classification issues [28]. The fundamental goal of NN is computing the distances globally between the competing patterns, and then ranking them to select the NN that characterizes a test pattern's class most accurately. Computing the distances between patterns is done by distance metrics and which can be degraded by noise and natural variability. If the feature vector is very large and if there exists a falsely assumed correlation, it may produce an irrelevant distance correlation. A rise in feature dimension causes a classification process to converge more slowly and erroneously [29]. The NN classifier is modified to address the aforementioned challenges and the classification of the test data is made by choosing the Nearest Features (NF) of the train data [30]. However, this method only works with feature vectors of the specified length. Due to these reasons, developing a classifier that deals with variable-length high-dimensional input is still an unresolved challenge in technology. In this paper, a novel Nearest Angle (NA) classification algorithm based on SURF descriptors and angles between the matching Interest Points is proposed. This method produces local-level similarity scores and rankings and finally generates the overall result of the classifier for a global decision on the test pattern.

SURF is a scale and rotation invariant feature detector and a simple and accurate descriptor with good repeatability, robustness, and distinctiveness [31]. The feature extraction process of SURF is composed of two steps. First, detect interest points (IP) from the meaningful structures of an image, which means finding the same physical structures under diverse viewing conditions. This method gives local features with the required level of invariance due to its focus on scale and rotation invariant feature detector and descriptor. These IPs which are characterized by feature descriptors are matched between different images. The search of correspondence requires the comparison of IPs in images where they are seen at different scales. The descriptor space of two images is considered at a time and the descriptor distances of all possible combinations of IP pairs between the images are calculated. These IP pairs along with a classification approach with a small number of parameters play a major role in the proposed method for the classification of the handwritten character recognition system that is considered for this study.

III. Proposed Methodology

In this section, we discuss the algorithms and methods involved in the proposed Nearest Angles (NA) classifier for handwritten character recognition. First, we resize all images to a uniform size of 256x256 pixels. Then, we extract SURF descriptors from each image. SURF descriptors are a type of feature that is robust to noise and changes in illumination. Next, we find the nearest interest point in each training image for each interest point in the test image using a minimum distance criterion. The crux of our method is to calculate the angle between each pair of corresponding interest points. We then use these angles to classify the test image by assigning it to the class of the training image with the nearest angle. To improve the accuracy of our classification, we use a "top-rank" technique that considers the highest-scoring training images to be more representative of their class. Finally, we use a global decision-making phase to assign the test image to one of the established classes. This phase involves systematic calculations and normalizations. The overall process flow of this proposed method is represented in Fig. 1.

A. *Nearest Angles (NA) from SURF Feature Detector and Descriptor*

A simple variant of the Nearest Neighbor classifier called the Nearest Angles (NA) classifier is proposed that is robust during recognition of the text recognition system, with the variable-length feature vector. The utilization of deep learning techniques and the complexities of computation brought on by larger dimensions and parameters can be minimized if an easy-to-use method is created to handle the IPs produced by SURF descriptors [32]. The proposed approach extends the classification decision at the local level and at the global level. This increases the chance of overcoming the misclassification of local classifiers. The following sections explain how to determine different NAs for local-level decisions and finally, the classification of the test data using the NA classifier is given in detail. As the number of SURF descriptors is large and varies in number from image to image, only a few of them are shown in Table I-II, to illustrate the proposed method.

B. *Determination of Matching Interest Point Pairs from SURF Descriptors*

The proposed method extracts Nearest Angles (NAs) from the SURF matching points between two images in the spatial coordinate domain. The previous approach, the NIP classifier, illustrates a sample training set and test image with few *IPs[32]*. The training data contains 3, 3 and two images from the classes C1, C2, and C3 respectively. The training data: $G \in \{C_{11}, C_{21}, C_{31}, C_{12}, C_{22}, C_{32}, C_{13}, C_{23}\}$. Let C_{jc} is the j^{th} image of class c. For each image, a different number of IPs are produced. Let In_{ijc} be the i^{th} IP of j^{th} object of the class c. Each IP is described by a local descriptor vector of fixed size. Two images are compared by computing the IP-to-IP distance between M IPs of each image C_{jc} *with M IPs and the test image Z.*

Evaluating the Distance between IPs

Considering the training data, where the IPs of each image with a class label c is $\{T = \{In_{pj_c}, \forall p, j, c \in [1, X], X \in$ Integer $\}$ and IPs of the image n with a label $\bar{c}$ is $\{H = \{In_{qn_{\bar{c}}}, \forall m, n, \bar{c} \in [1, X], X \in$ Integer $\}$ where $c \neq \bar{c}$ and $c, \bar{c} \in$ Class Label and p and q varies from 1 to number of IPs of images j and n respectively. Let us calculate the distance between two images j and n by calculating the distance between their IPs. In_{ij_c} is i^{th} IP of image j from class c and $In_{mn_{\bar{c}}}$ is m^{th} IP of image n from class $\bar{c}$. The distances among IPs In_{ij_c} and $In_{mn_{\bar{c}}}$ are computed using simple trigonometry [32]. The distance between two IPs In_{ij_c} and $In_{mn_{\bar{c}}}$ is calculated using Eq. (1).

$$d = \left|In_{ij_c} - In_{mn_{\bar{c}}}\right| / \sqrt{2} \quad (1)$$

The minimum distance is determined after repeating this method for every IPs of n in H and using Eq. (2).

$$di = min\{\left|In_{ij_c} - In_{qn_{\bar{c}}}\right| / \sqrt{2}\}, \quad (2)$$

where q ranges from 1 to the total number of IPs in the image n. The m^{th} *IP of n* $In_{mn_{\bar{c}}}$, with minimum distance di is taken to form a matching pair $(In_{ij_c}, In_{mn_{\bar{c}}})$. The meaning is i^{th} IP of image j from class produced minimum distance with m^{th} IP of image n from class $\bar{c}$. Repeat the above technique for all the IPs in set G to determine the matching pairs of *IPs* of image j (see Fig. 2).

Fig. 3(a) to Fig. 3(b) shows the images and IPs forming correct matching pairs (each straight line represents a line connecting the IPs of each matching pair at comparing images). The angle between the two points in the matching pair is computed by the principle of trigonometry as shown in Fig. 3(c) to Fig. 3(d). For every matching IP pair, four angles can be generated. But any one of the angles is sufficient to decide NAs. Fig. 3 shows the two Tamil character images ‘Ah’ and the points A and B are the IPs in a matching pair after applying the distance Eq. (2) over SURF descriptors in both images. Let PQ and RS be the lines drawn parallel to these points. The ∠ABR and ∠BAQ are alternate interior angles and are equal. As when parallel lines get crossed by another line, alternate interior angles are equal and are calculated as given in Definition 1. Similarly, ∠PAB and ∠ABS are also alternate interior angles.

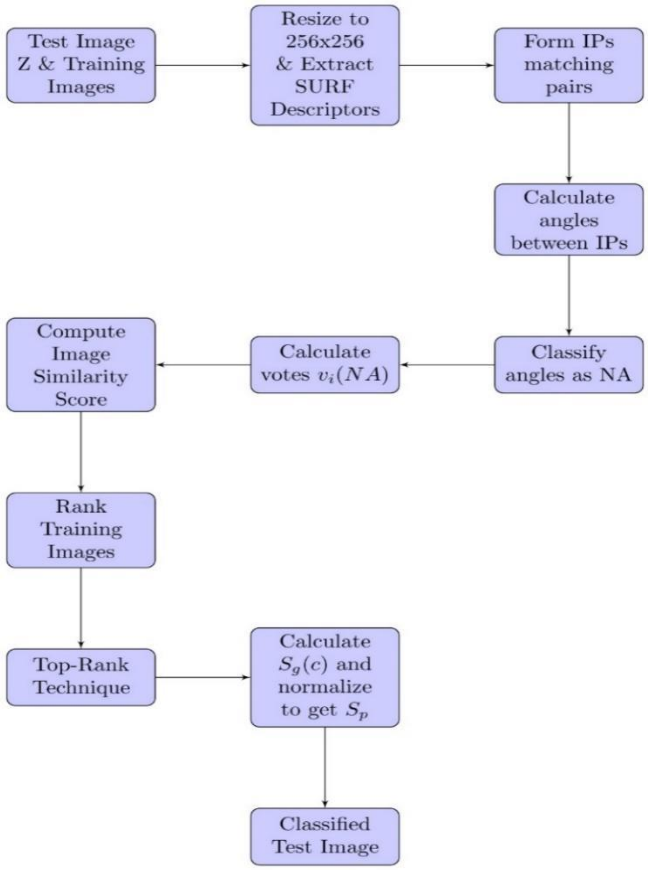


Fig. 1. Overall process flow of the proposed nearest angles (NA) classifier for handwritten character recognition.

Two values are obtained from each alternate interior angle, so four angles are obtained from each matching pair of IPs. Only one angle value is used for processing for each matching IP pair as the proposed method works for all four angles at the same time. Initially, the bounding box of both the test and training images are calculated and resized to 256 x 256, then SURF features are extracted and matching pairs are generated using the IPs of both the images used for comparison. A line is drawn to connect both the IPs of the matching pair. Parallel lines are drawn to detect the angles between the line drawn and the corresponding axis (see Definition 1).

DEFINITION 1: *When a line crosses parallel lines, alternate interior angles* $An = (\frac{y2-y1}{x2-x1}) * \frac{180}{\pi}$ *are equal where (x1, y1) and (x2, y2) are the coordinates of the line crossing the parallel lines.*

C. Determination of NAs

For each matching pair between the test and training image, two pairs of angles are produced as alternate interior angles as in Fig. 2(d) *∠PAB and ∠ABS are alternate interior angles and so ∠PAB =∠ABS. Similarly, ∠ABR and ∠BAQ are alternate interior angles and so ∠ABR = ∠BAQ). A*nd there are two distinct values out of four angles as alternate interior angles are equal. But, any one of the values out of two can be used for processing. Table I shows the angle generated between the IPs of test image *Z* and each of the training images (only one angle of particular IP matching pair is given). There are seven images in the training set and each image is with different number of IPs. The test image *Z* produces IPs matching pair with *IPs* of the training image, and the angles between IPs from the matching pair are calculated using Definition 1 and are shown in Table I. After determining the angle between IPs in the matching pair, the next step is to determine the NAs of each of the training images.

DEFINITION 2: *An angle* a_n *between the IPs from the test image and training image j in the matching pair is classified as one of the NAs of training image j if and only if* $|\theta_{lan}| \leq a_n \geq |\theta_{uan}|$ *where* θ_{lan} *and* θ_{uan} *are the upper and lower bounds of NA detection threshold.*

After calculating the angle a_n of IPs from each matching pair of the training image *j*, a comparison is made with the θ_{lan} and $\theta_{uan,}$ which are lower and upper bound NA detection threshold values respectively. When compared with the threshold values θ_{lan} and θ_{uan}, based on Definition 2, these angles give the measure of closeness of *IP* of test image *Z*, with the training image *j* and which is used to make a local decision using NAs. An angle a_n is said to be NA of training image *j*, if it is greater than θ_{lan} *and* less than θ_{uan}. The process of determining the NAs is given by voting as shown in Eq. (3).

$$vi(NA) = \begin{cases} 1, & if\ \theta_{lan} \leq a_n \leq \theta_{uan} \\ 0, & otherwise \end{cases} \quad (3)$$

If vi(NA)=1, the angle a_n is considered as NA between the training and test image.

D. Local Decision using NAs Classifier

The angle a_n from a training image is said to be the Nearest Angle (NA), if and only if $|\theta_{lan}| \leq |a_n| \leq |\theta_{uan}|$ where θ_{lan} and θ_{uan} are lower and upper bounds of NAs threshold values which are assumed to be 80 and 100 respectively. Each NA of the training image carries a vote *vi(NA)*of 1. For each of the training images, the image similarity score is calculated by summing up its *votes* on *NAs which are* shown in Table I.

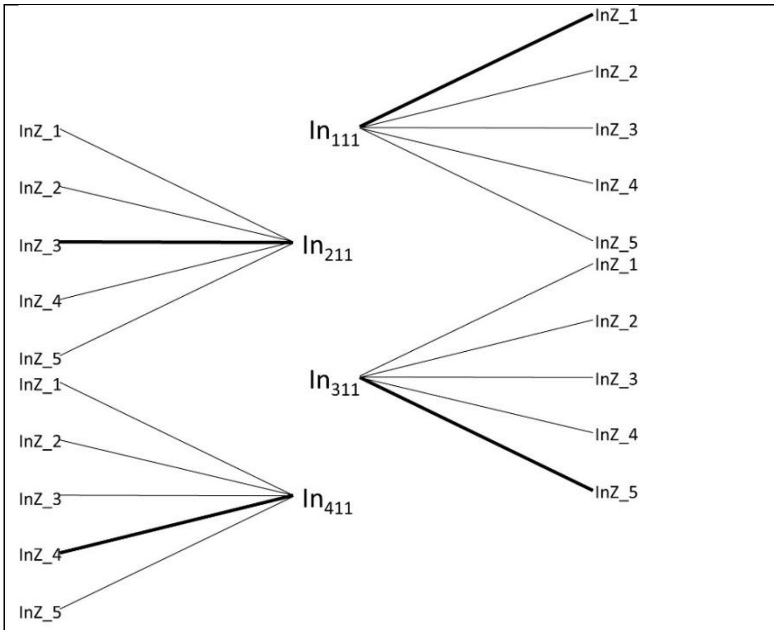
The training image C_{11} produces the matching pairs *($In_{111,}$ InZ_1), ($In_{211,}$ InZ_3), ($In_{311,}$ InZ_5) and ($In_{411,}$ InZ_4)* with test image *Z* and the angles $a_n$calculated are 80,110,102 and 96 respectively. The votes are generated by comparing the angles with the threshold values of NA θ_{lan} and θ_{uan}. Using (4) the Image similarity score for training image C_{11} is 2 as angles 80 and 96 are greater than the threshold θ_{lan} (i.e.,80) and are less than θ_{uan} (i.e.,100) and 110 and 102 are greater than θ_{uan} (i.e.,100). These image similarity scores at the local decision phase are maintained to determine the class of the test image at the global level.

TABLE I. THE PROPOSED NA VOTE AND IMAGE SIMILARITY SCORE COMPUTATION TO DETERMINE THE CLASS OF Z FROM THE TAMIL HANDWRITTEN CHARACTER DATABASE

IPs of the test image Z are $InZ_1, InZ_2, InZ_3, InZ_4$ and InZ_5													
Class-label	Object label	Matching-pair	Angle	Matching-pair	Angle	Matching-pair	Angle	Matching-pair	Angle	Matching-pair	Angle	Image similarity score based on NAs	Normalized image similarity score
C1	C_{11}	$(In111, InZ_1)$	80	$(In211, InZ_3)$	110	$(In311, InZ_5)$	1022	$(In411, InZ_4)$	96			2	2/4=.5
	vote	1		0		0		1					
	C_{21}	$(In121, InZ_2)$	96	$(In221, InZ_1)$	100	$(In321, InZ_3)$	89	$(In421, InZ_4)$	86	$(In521, InZ_5)$	93	5	5/5=1
	vote	1		1		1		1		1			
	C_{31}	$(In131, InZ_1)$	120	$(In231, InZ_2)$	89	$(In331, InZ_3)$	93	$(In431, InZ_5)$	85			3	¾=.75
	vote	0		1		1		1					
C2	C_{12}	$(In112, InZ_1)$	97	$(In212, InZ_4)$	110	$(In312, InZ_3)$	131	$(In412, InZ_5)$	142			1	¼=.25
	vote	1		0		0		0					
	C_{22}	$(In122, InZ_1)$	80	$(In222, InZ_3)$	95	$(In322, InZ_5)$	118	$(In422, InZ_2)$	94	$(In522, InZ_4)$	88	4	4/5=.8
	vote	1		1		0		1		1			
	C_{32}	$(In132, InZ_2)$	127	$(In232, InZ_1)$	122	$(In332, InZ_4)$	94	$(In432, InZ_3)$	107	$(In532, InZ_5)$	140	1	1/5=.2
	vote	0		0		1		0		0			
C3	C_{13}	$(In113, InZ_2)$	130	$(In213, InZ_3)$	136	$(In313, InZ_1)$	129					0	0
	votes	0		0		0							
	C_{23}	$(In123, InZ_1)$	98	$(In223, InZ_3)$	117	$(In323, InZ_2)$	82	$(In423, InZ_5)$	128	$(In523, InZ_4)$	132	2	2/5=.4
	votes	1		0		1		0		0			

TABLE II. DETERMINATION OF THE CLASS OF Z BASED ON GLOBAL DECISION

Collective class similarity scores, $S_g(c)$					Class probability score for each rank S_p			
	Rank				Rank			
Class, c	1	2	3		1	2	3	
C1	1	1.75	2.25		**1/1=1**	1.75/1.75=1	2.25/2.25=1	
C2	.8	1.05	1.25		.8/1=..8	1.05/1.75=.6	1.25/2.25=.55	
C3	.4	.4	-		.4/1=.4	.4/1.75=.228		



(a)

Matching pair (IP of image C11, IP of image Z)

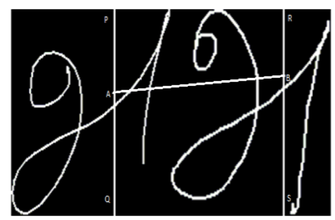
In_{111}	$In_{Z\,1}$
In_{211}	In_{Z-2}
In_{311}	$In_{Z\,5}$
In_{411}	$In_{Z\,4}$

(b)

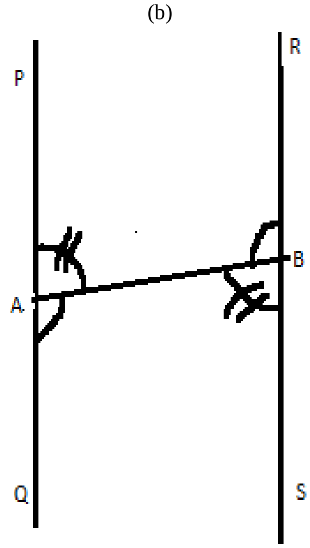
Fig. 2. (a). Matching pair formation between the IPs of C11 and Z. Solid line indicates the best match with minimum distance between IPs. (b) Best matches between IPs of C11 and Z.

(a)

(b)



(c)



(d)

Fig. 3. (a) and (b)Two intra-class Tamil Character images, ‘Ah’ and ‘AAh’, respectively, showing Matching pair with minimum distance. (c) Two Tamil Character images ‘Ah’ with matching IPs at points A and B. (d) The line connects the two points A and B and four angles can be produced between these two points, ∠PAB, ∠BAQ, ∠ABR and ∠ABS. (b)∠PAB and ∠ABS are alternate interior angles and so ∠PAB =∠ABS. Similarly, ∠ABR and ∠BAQ are alternate interior angles and so ∠ABR = ∠BAQ.

The Image similarity score gives the local decision on the closeness of IPs of the test data with training data.

E. Classification using Global Decision on Similarity Scores

The global decision on the recognition of test images is done by considering the ‘top-rank’ technique. Accordingly, the image in the training set, at each class, with the highest value in the ‘image similarity score’ is considered for further processing. In this method, the training image with the highest ‘Image similarity score’ in each class is considered to represent the whole class. That means, the training image in each class with a higher similarity value is calculated and is termed as collective class similarity score $S_g(c)$ at ‘Rank 1’ and is given in Table II. Class probability scores S_p is calculated by normalizing the collective class similarity score $S_g(c)$ by dividing it by the maximum similarity score across the classes. The highest value of Sp across the classes is considered as the class of the test image Z, c*, which can be denoted as Eq. (4).

$$c^* = arg\,max\ S_p(c) \qquad (4)$$

The top-rank approach generally gives many classes as outputs despite the fact that it facilitates decision-making at the global level. Therefore, the image similarity score at lower rankings is taken into account, and an algorithmic sum is performed at each ranking level. The image-level similarity scores at first and second ranks from each class are summed up to calculate the collective class similarity score $S_g(c)$ at Rank 2. This process is repeated for the subsequent lower-ranking levels. The rank-wise normalization *of* $S_g(c)$ throughout all the classes yields *class probability scores,* S_p. A tie-in-the-top-rank method is solved by adding the S_p values at the lower rankings. It would eliminate the ambiguity in class assignments caused by ties. The rank 1 S_p score of C1 yields the highest value of 1 with no ties (see Table II). The model is evaluated up to three ranking levels since an acceptable recognition is obtained close to rank 3 in the NA classification.

F. Overall Process Flow of Proposed NA Classifier

The Nearest Angles (NA) Classifier for Text Recognition using SURF Descriptors is a novel algorithm designed to address the challenges of text classification in images. It leverages Speeded-Up Robust Features (SURF) to capture unique descriptors and interest points from a set of pre-labelled training images across multiple categories. This approach enhances the conventional techniques by employing angle-based metrics to establish the nearest matches between the features of a test image and those in the training dataset. The algorithm then computes similarity scores based on these angle metrics to classify the test image into one of the predefined categories. The efficacy of the method lies in its ability to deliver accurate classifications while effectively handling variations in scale, orientation, and illumination. The proposed Nearest Angles (NA) Classifier's process flow is explained in Algorithm 1.

Algorithm 1 NA Classifier for Text Recognition

Input:
Training image set G with class labels $C1, C2, ..., Cn$
Test image Z
Lower bound NA detection threshold θ_{lan}
Upper bound NA detection threshold θ_{uan}
Output:
Classified label for the test image Z
Steps:
SURF Feature Extraction
for each image C_{jc} in G and test image Z **do**
Extract SURFdescriptors and Ips (IP_{ijc} for C_{jc} and IP_{zn} for Z)
end for
Calculate IP-to-IP Distances
for each IP_{ijc} and IP_{zn} **do**
$d = \frac{|IP_{ijc} - IP_{zn}|}{\sqrt{2}}$
end for
Find Minimum Distances
for each IP_{ijc} **do**
$d_i = \min\left(\frac{|IP_{ijc} - IP_{zn}|}{\sqrt{2}}\right)$
end for
Determine Matching Pairs
for each IP_{ijc} **do**
    Pair IP_{ijc} and IP_{zn} if d_i is minimum
end for
Calculate Angles for Matching Pairs
for each matching pair **do**
    $Angle(An) = \frac{(y_2 - y_i)}{(x_2 - x_i)} \times \frac{180}{\pi}$
end for
Determine Nearest Angles (NAs)
for each angle a_n **do**
    if $\theta_{lan} \leq a_n \leq \theta_{uan}$ **then**
        v_i(NA) = 1
    else
        v_i(NA) = 0
    end if
end for
Compute Image Similarity Score
for each image C_{jc} **do**
    Image Similarity Score = $^{\mathrm{P}}v_i$(NA)
end for
Local Decision
Image similarity scores of m top-ranked images are taken to obtain local decisions at 2 NAs.
Global Decision
Calculate class similarity score followed by class probability score to make a more robust global decision

The Nearest Angles (NA) Classifier for Text Recognition using SURF Descriptors is an algorithm designed to classify text images into pre-defined categories. The algorithm starts by extracting Speeded-Up Robust Features (SURF) descriptors and interest points (IPs) from both the training image set, labelled with classes $C1, C2, \ldots, Cn$, and a test image Z. It calculates the IP-to-IP distances between these features and identifies the minimum distances to form matching pairs of IPs between each training image and the test image. Angles are calculated for these matching pairs, and a "Nearest Angle" (NA) is identified based on lower and upper angle thresholds θ_{lan} and θ_{uan}. An image similarity score is computed for each training image based on these NAs. The algorithm then classifies the test image Z into the category of the training image with the highest similarity score. Optionally, a global decision can be made by aggregating multiple local decisions.

IV. Results and Discussion

A. System Setup

To evaluate the effectiveness of the NA Classifier, the experiments were conducted on a system equipped with an Intel i7 processor, 16 GB of RAM, and a 1 TB SATA hard drive. The system also featured an NVIDIA GPU. The software environment was MATLAB 2020, specifically utilizing its Deep Learning Toolbox, all running on a Windows 10 operating system**.**

B. Experimental Setup

The primary objective of this section is to assess the effectiveness of the proposed NA Classifier for handwritten text recognition, which is designed to reduce the computational burden commonly encountered in existing AI algorithms. The experimental framework is divided into three key components: computational efficiency, accuracy and comparative analysis: Intra-Class vs. Inter-Class Matching. These aspects are rigorously evaluated against a range of state-of-the-art algorithms and models. In the first component, the computational efficiency of the NA Classifier is analyzed to determine whether it effectively reduces processing time without sacrificing performance. In the second component, the method's error rate is examined to assess its reliability in various classification tasks. Finally, the accuracy of the NA Classifier is evaluated, serving as the critical measure of its overall effectiveness. The accuracy of the NA Classifier was evaluated against state-of-the-art methods, including: SurfCNN (A. M. Elmoogy et al. [20]), CNN-based Models (M. Rai and P. Rivas [19]), SURF with SVM (Shagun Katoch et al. [21]), SVD with SVM (Li C et al. [23]), VGG16 and MobileNetV2 (Ardiant Utomo et al. [27]), Nearest Neighbor (Nitin Bhatia, Vandana [28]), Modified-GA (Ashlin Deepa R N, Rajeswara Rao R [32]), Nearest Angle (NA) Algorithm (Alex Pappachen James [30]). The proposed method was also benchmarked against the popular machine learning and deep learning algorithms, including k-nearest neighbors (k-NN), k-means clustering, support vector machines (SVM), decision trees, Naive Bayes, Random Forest, VGG-19, MobileNet, and gradient boosting.

C. Dataset Details

In this research, we employ three distinct Indic handwriting datasets sourced from HP Lab, focusing on the Tamil, Devanagari, and Telugu scripts. These datasets serve as the foundational testing ground for the evaluation of our proposed method, encompassing a wide array of handwritten textual samples in these languages to validate the robustness and applicability of our approach. Tables III, IV, and V provide details about the Tamil, Telugu, and Devanagari datasets, respectively.

TABLE III. Tamil Dataset Attribute Details

Criteria	Details
Total Classes	156
Data Format	UNIPEN v1.0, bilevel TIFF images
Training Set	70% of samples from each class, randomly selected
Test Set	Remaining 30% of samples from each class

TABLE IV. Telugu Dataset Attribute Details

Criteria	Details
Total Classes	166
Data Format	UNIPEN v1.0
Training Set	70% of samples from each class, randomly selected
Test Set	Remaining 30% of samples from each class

TABLE V. Devanagari Dataset Attribute Details

Criteria	Details
Total Classes	111
Data Format	UNIPEN v1.0, bilevel TIFF images
Training Set	70% of samples from each class, randomly selected
Test Set	Remaining 30% of samples from each class

D. Computational Efficiency Analysis

Computational efficiency is a critical criterion in the evaluation of deep learning and machine learning algorithms. In recent years, there has been significant research on mitigating the computational overhead associated with deep learning and machine learning techniques. This section evaluates the computational efficiency of the proposed NA classifier on three handwritten datasets: Tamil, Telugu, and Devanagari. For each dataset, 70% of the samples from each class were allocated to the training set, and the remaining 30% constituted the test set. First, we evaluate the training efficiency of the proposed method against state-of-the-art handwritten character recognition methods, deep learning algorithms, and machine learning algorithms.

According to the Fig. 4, the proposed Nearest Angle (NA) classifier demonstrates a notable advantage in computational efficiency across three handwritten datasets: Tamil, Telugu, and Devanagari, with training times of 3.0, 3.2, and 3.4, respectively. These times are significantly lower than those of existing state-of-the-art methods and standard machine learning and deep learning algorithms, such as VGG-19, MobileNet, and SVM, which have training times up to three times higher. This reduced training time implies a lower computational burden, fewer resource requirements, and faster deployment capabilities, thereby making the NA classifier an ideal choice for applications where computational resources and time are critical constraints.

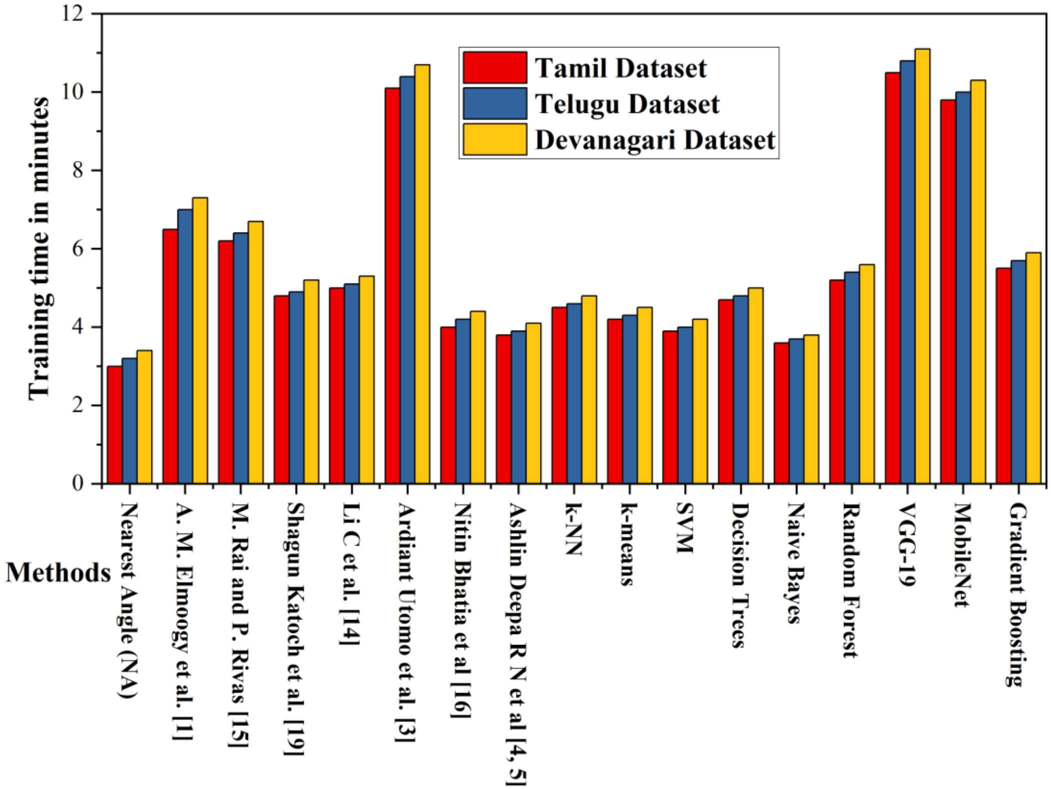


Fig. 4. Comparative analysis of training time across various algorithms and datasets.

To assess the classification time efficiency of the proposed method in a rigorous manner, we adopted a systematic testing approach. Specifically, we randomly selected five samples from each of three different datasets. Each set of five samples was classified using the proposed method and various existing methods for comparison. For each set, we measured the time required to complete the classification and then calculated the average classification time for that set. Finally, we computed an overall average classification time across the three sets to obtain a robust estimate of performance.

The Fig. 5 comparing average classification times reveals that the proposed NA method is the most time-efficient with an average of 0.15 seconds, closely followed by the method proposed by A. M. Elmoogy et al. at 0.18 seconds. Traditional machine learning algorithms like k-NN, k-means, and SVM exhibit moderate speed, ranging from 0.29 to 0.38 seconds, while ensemble methods such as Random Forest and Gradient Boosting are slightly slower within the same category. Notably, deep learning models like VGG-19 and MobileNet are significantly slower, taking around 0.95 and 0.90 seconds respectively, likely due to their complex architectures and higher computational requirements.

Compared to traditional deep learning methods, the proposed NA classifier for text recognition systems significantly reduces classification and training time. This efficiency is largely due to its localized decision-making approach, which focuses on calculating angles between matching Interest Points (IPs) in the spatial coordinate domain. Instead of computing high-dimensional feature vectors and performing complex operations, the NA classifier uses simple calculations to evaluate the closeness of IPs. By considering only angles that fall within predefined upper and lower bound thresholds, the system effectively performs dimensionality reduction. This limits the computational complexity, which is especially beneficial when dealing with large sets of SURF descriptors that can vary in number across images. As a result, this method not only minimizes the chances of misclassification at the local level but also reduces computational overhead, leading to faster classification and training times.

E. Accuracy Analysis

Accuracy is a crucial metric for evaluating the performance of the NA Classifier in handwritten text recognition. Specifically, we are interested in the algorithm's effectiveness in identifying individual characters in three datasets: Tamil, Telugu, and Devanagari. In this section, we will outline the metrics used to assess accuracy and present the results in comparison with state-of-the-art models and algorithms. The following metrics are primarily involved in the evaluation of accuracy:

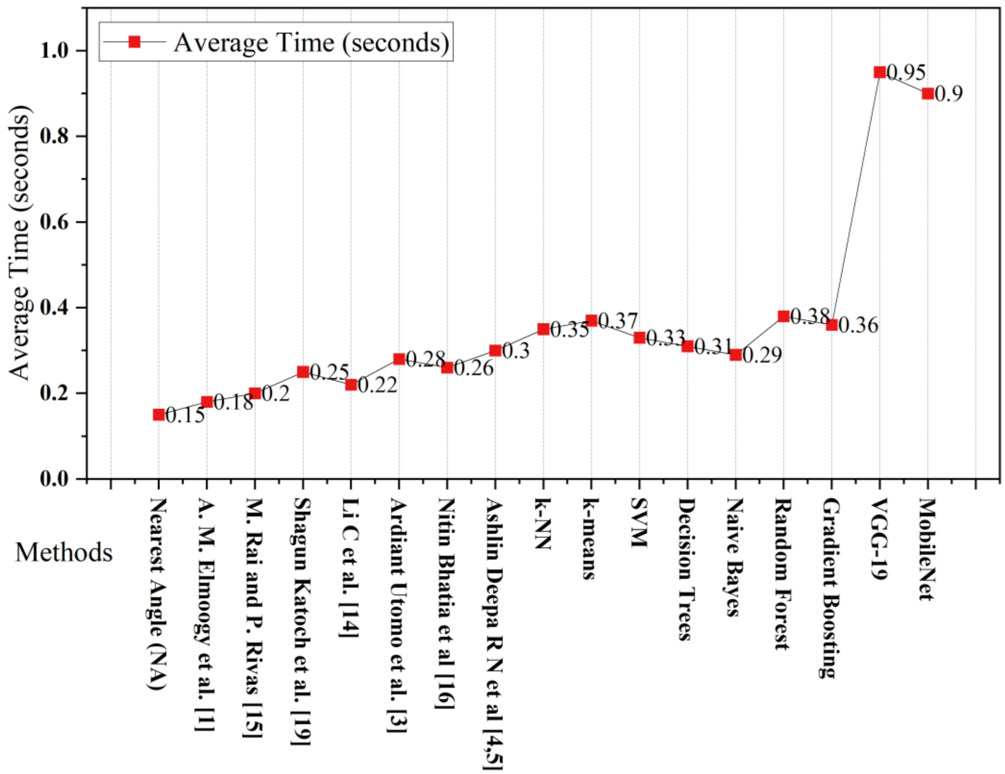


Fig. 5. Average classification time in seconds.

1) True positives (TP): These represent instances where the NA Classifier accurately identifies a character as belonging to a specific class within the dataset.

2) True negatives (TN): These denote situations where the NA Classifier accurately concludes that a given sample does not belong to a targeted class.

3) False positives (FP): These occur when the NA Classifier incorrectly ascribes a sample to a particular class when it should not have.

4) False negatives (FN): These represent the scenarios where the NA Classifier fails to identify a sample as belonging to a specific class when it actually does. Based on these, we calculate the following metrics: Accuracy (ACC), Sensitivity (SEN), Specificity (SPEC), F1-Score, and Matthews Correlation Coefficient (MCC) which are calculated by using following formulas.

$$ACC = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (5)$$

$$SEN = \frac{TP}{(TP + FN)} \quad (6)$$

$$SPEC = \frac{TN}{(TN + FP)} \quad (7)$$

$$F1 = 2 * \frac{(precision * recall)}{(precision + recall)} \quad (8)$$

TABLE VI. BENCHMARKING ACCURACY OF NA CLASSIFIER AGAINST STATE-OF-THE-ART METHODS ON TAMIL DATASET

Methods	ACC	SEN	SPEC	F1-Score
Nearest Angle (NA)	96.4%	95.8%	97.3%	96.0%
A. M. Elmoogy et al. [20]	90.2%	89.5%	91.1%	90.0%
M. Rai and P. Rivas [19]	89.8%	88.0%	90.9%	89.5%
Shagun Katoch et al. [21]	88.5%	87.6%	90.0%	88.0%
Li C et al. [23]	87.0%	86.1%	88.5%	86.5%
Ardiant Utomo et al. [27]	92.3%	91.0%	94.0%	92.0%
Nitin Bhatia et al [28]	85.0%	83.5%	86.8%	84.0%
Ashlin Deepa R N et al [32]	83.7%	82.9%	85.2%	83.0%
k-NN	82.0%	81.5%	84.0%	82.0%
k-means	79.0%	78.5%	81.0%	79.0%
SVM	81.5%	80.0%	83.5%	81.0%
Decision Trees	77.5%	76.0%	80.0%	77.0%
Naive Bayes	76.0%	74.8%	78.5%	75.0%
Random Forest	80.5%	79.0%	83.0%	80.0%
Gradient Boosting	81.0%	79.5%	84.0%	80.5%
VGG-19	97.2%	96.8%	98.0%	97.0%
MobileNet	97.5%	97.0%	98.2%	97.3%

TABLE VII. BENCHMARKING ACCURACY OF NA CLASSIFIER AGAINST STATE-OF-THE-ART METHODS ON TELUGU DATASET

Methods	ACC	SEN	SPEC	F1-Score
Nearest Angle (NA)	96.5%	95.8%	97.2%	96.1%
A. M. Elmoogy et al. [20]	94.2%	93.0%	95.5%	94.3%
M. Rai and P. Rivas [19]	93.8%	92.7%	94.9%	93.8%
Shagun Katoch et al. [21]	92.5%	91.2%	93.8%	92.5%
Li C et al. [23]	91.8%	90.1%	93.5%	91.8%
Ardiant Utomo et al. [27]	97.2%	96.9%	97.5%	97.2%
Nitin Bhatia et al [28]	89.5%	88.2%	90.8%	89.5%
Ashlin Deepa R N et al [32]	88.3%	87.0%	89.6%	88.3%
k-NN	90.0%	88.8%	91.2%	90.0%
k-means	87.2%	86.0%	88.4%	87.2%
SVM	91.5%	90.0%	93.0%	91.5%
Decision Trees	85.8%	84.2%	87.4%	85.8%
Naive Bayes	84.5%	82.7%	86.3%	84.5%
Random Forest	89.0%	87.8%	90.2%	89.0%
Gradient Boosting	90.5%	89.2%	91.8%	90.5%
VGG-19	97.5%	97.0%	97.2%	96.5%
MobileNet	97.1%	96.7%	97.3%	97.2%

TABLE VIII. BENCHMARKING ACCURACY OF NA CLASSIFIER AGAINST STATE-OF-THE-ART METHODS ON DEVANAGARI DATASET

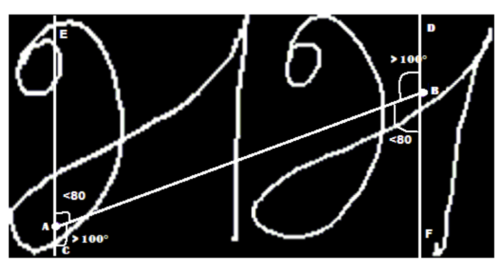
Methods	ACC	SEN	SPEC	F1-Score
Nearest Angle (NA)	97.0%	96.4%	97.6%	96.7%
A. M. Elmoogy et al. [20]	95.2%	94.0%	96.4%	95.2%
M. Rai and P. Rivas [19]	94.5%	93.5%	95.5%	94.5%
Shagun Katoch et al. [21]	93.0%	91.9%	94.1%	93.0%
Li C et al. [23]	92.1%	90.8%	93.4%	92.1%
Ardiant Utomo et al. [27]	97.8%	97.5%	98.1%	97.8%
Nitin Bhatia et al [28]	90.5%	89.2%	91.8%	90.5%
Ashlin Deepa R N et al [32]	89.0%	87.8%	90.2%	89.0%
k-NN	91.0%	89.7%	92.3%	91.0%
k-means	87.5%	86.3%	88.7%	87.5%
SVM	92.5%	91.0%	94.0%	92.5%
Decision Trees	86.2%	84.6%	87.8%	86.2%
Naive Bayes	85.0%	83.3%	86.7%	85.0%
Random Forest	89.5%	88.1%	90.9%	89.5%
Gradient Boosting	90.8%	89.5%	92.1%	90.8%
VGG-19	98.7%	98.2%	99.2%	98.7%
MobileNet	97.3%	97.0%	98.6%	97.3%

According to the data in Tables VI, VII, and VIII, the NA classifier consistently outperforms traditional machine learning algorithms such as k-NN, k-means, SVM, and Decision Trees across all datasets. In the Tamil dataset, for example, the NA classifier achieved an accuracy of 96.4%, which is notably higher than the closest traditional competitor, Ardiant Utomo et al, at 92.3% (see Table VI). Similar trends are observed in the Telugu and Devanagari datasets as well, where the NA classifier scored 96.5% and 97.0% respectively. In terms of sensitivity and specificity, the NA classifier also performs excellently. It managed to attain 95.8% sensitivity and 97.3% specificity on the Tamil dataset, again superior to any traditional algorithm (see Table VI). The high sensitivity and specificity scores mean that the NA classifier is proficient at correctly identifying true positives and true negatives, making it a reliable choice for real-world applications. The F1-Score serves as a balanced measure of a model's performance, taking into account both precision and recall. The NA classifier achieves a high F1-Score of 96.0%, 96.1%, and 96.7% on the Tamil, Telugu, and Devanagari datasets, respectively as shown in Table VI, Table VII and Table VIII. This is a significant achievement compared to traditional classifiers. While deep learning methods like VGG-19 and MobileNet slightly outperform the NA classifier in terms of accuracy, they do so at the cost of computational resources. The NA classifier is designed for lightweight applications and is three times more computationally efficient than these deep learning models. In scenarios where computational resources are a concern, this efficiency makes the NA classifier an appealing choice without significantly compromising accuracy.

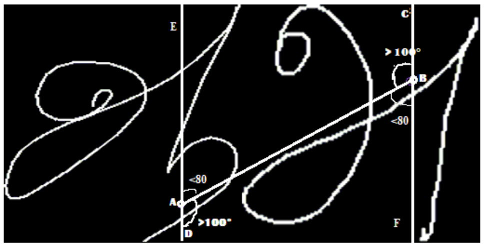
The NA classifier for handwritten character recognition that is both accurate and efficient. It is not as accurate as deep learning models, but it is much faster. This makes it a good choice for applications where speed is important, such as real-time text recognition on mobile devices. The NA classifier has been shown to work well on multiple datasets, which shows that it is reliable and robust.

F. Comparative Analysis: Intra-Class vs. Inter-Class Matching

Intra-class and inter-class matching discrimination is a crucial component of image matching in our study. Intra-class matching refers to matching pairs of data points that belong to the same class. Inter-class matching refers to matching pairs of data points that belong to different classes. Fig. 6(a) shows an example of intra-class matching. Both IPs A and B point to the same local physical structures in the two Tamil character images of 'Ah'. However, the NA classifier generates a vote of '0' because the alternate interior angles violate the lower and upper bounds of the NA detection thresholds. This means that the two images are not considered to be NAs. Fig. 6(b) shows an example of inter-class matching. The two IPs point to different local physical structures in the two character class images. The NA classifier also generates a vote of '0' in this case because the alternate interior angles violate the rule of NAs.

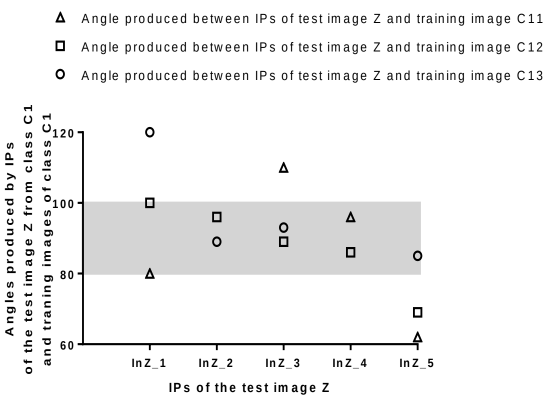


(a)



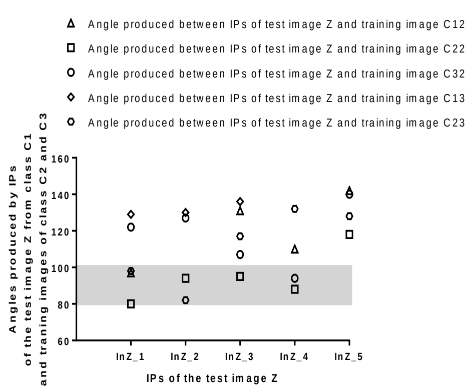
(b)

Fig. 6. (a) Two intra-class images of Tamil alphabet ‘Ah’ showing incorrect IPs matching at A and B, so that the angle ∠BAC and ∠ABD have more than 100° and the angle ∠EAB and ∠ABF have less than 80°. (b) Two inter-class images of Tamil alphabet ‘AAh’ and ‘Ah’ showing incorrect IPs matching at A and B, so that the angle ∠BAD and ∠ABC have more t100° and the angle ∠EAB and ∠ABF have less than 80°.



(a)

(b)



(b)

Fig. 7. Effect of parameters used in the NA classifier. (a) Most of the angles produced between IPs of test image Z and training images of class C1 are inside the shaded region which illustrates the allowed boundaries of NA classifier. (b) Most of the angle s produced between IPs of test image Z and training images of class C2 and C3 are outside the shaded region which illustrates the allowed boundaries of NA classifier.

Fig. 7(a) shows the reliability of the threshold values θuan=100 and θlan=80 for intra-class matching. Most of the IPs between the test image Z of class C1 and the intra-class training images C11, C21, and C13 produce angles within the upper and lower bounds. This means that the test image Z is correctly classified as not being an NA. Fig. 7(b) shows the angles formed between inter-class training images and the test image Z. Most of the angle values are out of the upper and lower bounds of the threshold values. This means that the test image Z is correctly classified as being an NA. The comparative analysis of intra-class vs. inter-class matching for NA detection shows that the NA classifier can produce good accuracy on the datasets used for the study, even though SURF features are scale and rotation-invariant. This is because the NA classifier uses additional information about the geometry of the local physical structures to detect NAs.

V. Discussion

The Nearest Angle (NA) classifier demonstrates promising performance for handwritten text recognition (HTR) tasks across a diverse range of datasets, including Tamil, Telugu, and Devanagari. Several key observations can be made from the study:

Superiority over Traditional Classifiers: The NA classifier consistently outperforms traditional machine learning methods such as k-N), k-means clustering, SVM, and decision trees in terms of accuracy, sensitivity, specificity, and F1-score. This highlights its potential as a robust and efficient classifier for HTR tasks.

Comparison with Deep Learning Models: While deep learning architectures such as VGG-19 and MobileNet marginally surpass the NA classifier in terms of accuracy, they do so at the expense of computational efficiency. The NA classifier's rapid classification time presents an optimal trade-off between speed and accuracy, especially critical for resource-constrained devices and real-time applications.

Reliability across Datasets: The NA classifier's consistent performance across Tamil, Telugu, and Devanagari datasets underscores its generalizability. This versatility is essential for a classifier, especially in applications where diverse scripts might be encountered.

Intra-class vs. Inter-class Matching: The comparative analysis on intra-class and inter-class matching reinforces the NA classifier's robustness. By exploiting the geometry of local physical structures, the NA classifier effectively differentiates between characters that may look visually similar, thereby minimizing false matches.

Thresholding in NA: The use of threshold values (θuan=100 and θlan=80) for angle classification serves as a vital mechanism for the NA classifier. As the presented figures depict, the chosen thresholds enable effective discrimination between intra-class and inter-class matches. The robustness of these thresholds in classifying the characters accurately showcases the model's resilience to typical variances seen in handwritten data.

Potential for Real-World Applications: Given its quick classification time, combined with high accuracy, the NA classifier is particularly suited for real-world scenarios, especially those demanding swift recognition such as mobile OCR applications or real-time transcription services.

VI. CONCLUSION

Local feature detectors and descriptors are powerful in many object recognition tasks. The complexity of deep-learning-based approaches can be avoided by using effective local descriptors such as SURF for feature extraction. In this study, the concept of local feature detection along with a similarity-voting-based classifier is introduced at the local level and a ranking-based classifier at the global level. The proposed Nearest Angles classifier is applied effectively with SURF descriptors for handwritten character recognition of Tamil, Devanagari, and Telugu scripts. With a slight compromise in the recognition accuracy, the proposed method facilitates the features of variable length as well as high dimensionality and reduces complexity. Benchmark databases are used to demonstrate robust recognition performance.

The usage of the collective class similarity score produces excellent results in the NA classifier as the false similarities are removed by selecting the IPs based on the angles between them. NA handles high dimensionality and variable-length feature vectors (IPs of training images). The study proves that local feature descriptors extract a huge number of informative features and can produce good results if used wisely with an efficient classifier. The proposed method utilizes local decision-making to reduce the computational overhead and achieves high classification accuracy rates of 91.0% for Tamil, 94.7% for Devanagari, and 88% for Telugu handwritten character datasets.

The capabilities of HCR systems can be improved in several ways. Transfer learning can be used to transfer knowledge from one script to another, which can reduce the need for large script-specific datasets. Additionally, HCR systems can be adapted for online recognition, allowing for real-time processing and feedback. This would be useful for applications such as digital signature verification and interactive systems. These enhancements would make HCR systems more adaptable, precise, and user-friendly, which would expand their usefulness in a variety of domains.

ACKNOWLEDGMENT

This work was supported by the Department of Science and Technology (DST), New Delhi, for the Research project grant Sanction Number: SP/YO/2021/2096 (C & G).

REFERENCES

[1] J. Nockels, P. Gooding, S. Ames, and M. Terras, “Understanding the application of handwritten text recognition technology in heritage contexts: a systematic review of Transkribus in published research,” Archival Science, vol. 22, no. 3, pp. 367–392, Jun. 2022.

[2] A. Chaudhuri, K. Mandaviya, P. Badelia, and S. K Ghosh, “Optical Character Recognition Systems for Different Languages with Soft Computing”, Cham: Springer International Publishing, pp. 1-7, 2017.

[3] R. Shah, M. K. Gupta and A. Kumar, "Ancient Sanskrit Line-level OCR using OpenNMT Architecture," 2021 Sixth International Conference on Image Information Processing (ICIIP), Shimla, India, pp. 347-352, 2021.

[4] C. Vasantha Lakshmi and C. Patvardhan, "A high accuracy OCR system for printed Telugu text," TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region, Bangalore, India, vol. 2, pp. 725-729, 2003.

[5] R. Saluja, D. Adiga, P. Chaudhuri, G. Ramakrishnan and M. Carman, "Error Detection and Corrections in Indic OCR Using LSTMs," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, pp. 17-22, 2017.

[6] S. J. Basha, D. Veeraiah, G. Pavani, S. T. Afreen, P. Rajesh and M. S. Sasank, "A Novel Approach for Optical Character Recognition (OCR) of Handwritten Telugu Alphabets using Convolutional Neural Networks," 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, pp. 1494-1500, 2021.

[7] C. S. Anoop and A. G. Ramakrishnan, "Meta-Learning for Indian Languages: Performance Analysis and Improvements With Linguistic Similarity Measures," in IEEE Access, vol. 11, no. 8, pp. 82050-82064, 2023.

[8] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, , M.A Fadhel, , M. Al-Amidie, L. Farhan, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” Journal of Big Data, vol. 8, no. 1, pp. 1-74. 2021.

[9] L. Chen, S. Li, Q. Bai, J. Yang, S. Jiang, and Y. Miao, “Review of Image Classification Algorithms Based on Convolutional Neural Networks,” Remote Sensing, vol. 13, no. 22, pp. 4712, 2021.

[10] I. H. Sarker, “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions,” SN Computer Science, vol. 2, no. 6, pp. 1-20, 2021.

[11] I. Rodriguez-Conde, C. Campos, and F. Fdez-Riverola, “Optimized convolutional neural network architectures for efficient on-device vision-based object detection,” Neural Computing and Applications, vol. 33, no. 23, pp. 10469–10501, 2021.

[12] C. Chen, P. Zhang, H. Zhang, J. Dai, H. Zhang, Y. Zhang, “Deep Learning on Computational-Resource-Limited Platforms: A Survey,” Mobile Information Systems, vol. 2020, no. 6, pp. 1–19, 2020.

[13] S. Zhong, Y. Liu, and Q. Chen, “Visual orientation inhomogeneity based scale-invariant feature transform,” Expert Systems with Applications, vol. 42, no. 13, pp. 5658–5667, 2015.

[14] . Guo, R. Xie and G. Jin, "An Efficient Method for NMR Data Compression Based on Fast Singular Value Decomposition," in IEEE Geoscience and Remote Sensing Letters, vol. 16, no. 2, pp. 301-305, 2019.

[15] J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, “Deep learning in computer vision: A critical review of emerging techniques and application scenarios,” Machine Learning with Applications, vol. 6, pp. 100134, Aug. 2021.

[16] B. Shah and H. Bhavsar, “Time Complexity in Deep Learning Models,” Procedia Computer Science, vol. 215, pp. 202–210, 2022.

[17] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, “A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning,” Archives of Computational Methods in Engineering, vol. 26, no. 3, pp. 1071–1092, 2019.

[18] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” Artificial Intelligence Review, vol. 53, pp. 5455–5516, 2020.

[19] M. Rai and P. Rivas, "A Review of Convolutional Neural Networks and Gabor Filters in Object Recognition," 2020 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, pp. 1560-1567, 2020.

[20] A. M. Elmoogy, X. Dong, T. Lu, R. Westendorp and K. R. Tarimala, "SurfCNN: A Descriptor Accelerated Convolutional Neural Network for Image-Based Indoor Localization," in IEEE Access, vol. 8, pp. 59750-59759, 2020.

[21] S. Katoch, V. Singh, and U. S. Tiwary, “Indian Sign Language recognition system using SURF with SVM and CNN,” Array, vol. 14, pp. 100141, 2022.

[22] H. Bay, T. Tuytelaars, and Luc Van Gool, “SURF: Speeded Up Robust Features,” in Computer Vision – ECCV 2006, pp. 404–417, 2006.

[23] C. Li, L. Khan, and B. Prabhakaran, “Feature Selection for Classification of Variable Length Multiattribute Motions,” Springer eBooks, pp. 116-137, 2007.

[24] S. Bandyopadhyay, C. A. Murthy, and S. K. Pal, “Pattern classification using genetic algorithms: Determination of H,” Pattern Recognition

Letters, vol. 19, no. 13, pp. 1171–1181, 1998.

[25] A. D. R. Nelson and R. R. Rao, "A modified GA classifier for offline Tamil handwritten character recognition," International Journal of Applied Pattern Recognition, vol. 4, no. 1, pp. 89, 2017.

[26] H. Kim, E. Ahn, M. Shin, and S.-H. Sim, "Crack and Noncrack Classification from Concrete Surface Images Using Machine Learning," Structural Health Monitoring, vol. 18, no. 3, pp. 725–738, 2018.

[27] A. Utomo, E. F. Juniawan, V. Lioe, and D. D. Santika, "Local Features Based Deep Learning for Mammographic Image Classification: In Comparison to CNN Models," Procedia Computer Science, vol. 179, pp. 169–176, 2021.

[28] N. Bhatia, "Survey of Nearest Neighbor Techniques," IJCSIS) International Journal of Computer Science and Information Security, vol. 8, no. 2, pp.302-305, 2010.

[29] M. E. Houle, H. Kriegel, Peer Kröger, E. Schubert, and A. Zimek, "Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?," Lecture Notes in Computer Science, pp. 482–500, 2010.

[30] A. P. James and S. Dimitrijev, "Nearest Neighbor Classifier Based on Nearest Feature Decisions," The Computer Journal, vol. 55, no. 9, pp. 1072–1087, Feb. 2012.

[31] H. Bay, T. Tuytelaars, and Luc Van Gool, "SURF: Speeded Up Robust Features," in Computer Vision – ECCV 2006, pp. 404–417, 2006.

[32] R. N. Ashlin Deepa and R. Rajeswara Rao, "A novel nearest interest point classifier for offline Tamil handwritten character recognition," Pattern Analysis and Applications, vol. 4, no. 23, pp. 199–212, 2020.