

Nearest quad-tree (NQT) classifier: a novel framework for handwritten character recognition

Ashlin Deepa R. N., Prasad Chetti, P. Chandra Sekhar Reddy, Sorabh Lakhnupal, Abhishek Joshi & Soloveva O. V

To cite this article: Ashlin Deepa R. N., Prasad Chetti, P. Chandra Sekhar Reddy, Sorabh Lakhnupal, Abhishek Joshi & Soloveva O. V (2024) Nearest quad-tree (NQT) classifier: a novel framework for handwritten character recognition, Cogent Engineering, 11:1, 2347363, DOI: [10.1080/23311916.2024.2347363](https://doi.org/10.1080/23311916.2024.2347363)

To link to this article: <https://doi.org/10.1080/23311916.2024.2347363>



© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group



Published online: 10 May 2024.



Submit your article to this journal [↗](#)



Article views: 101



View related articles [↗](#)



View Crossmark data [↗](#)

Nearest quad-tree (NQT) classifier: a novel framework for handwritten character recognition

Ashlin Deepa R. N.^a , Prasad Chetti^b, P. Chandra Sekhar Reddy^a, Sorabh Lakhanpal^c, Abhishek Joshi^d and Soloveva O. V^e

^aDepartment of Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, India; ^bSchool of Computer Science and Information Systems, Northwest Missouri State University, Maryville, MO, USA; ^cSchool of Pharmacy, Lovely Professional University, Phagwara, India; ^dExecutive Director- Students Affairs & IT Services, Uttarakhand University, Dehradun, India; ^eInstitute of Heat Power Engineering, Kazan State Power Engineering University, Kazan, Russia

ABSTRACT

Handwritten character recognition (HCR) is a critical component of diverse applications, especially in multilingual regions such as India. Despite technological advancements, recognizing handwritten characters remains challenging because of variations in writing style, shape and linguistic nuances. This study presents an efficient HCR system tailored for Indian languages, such as Tamil and Telugu scripts, to address these challenges. The proposed nearest quad-tree (NQT) classifier enhances recognition accuracy while ensuring computational efficiency. This novel approach, a light-weight design, combines local feature-level decisions with a global decision-making process, thereby making it suitable for devices with limited computational resources. Key contributions include the emphasis of the NQT classifier on efficiency and lightweight design, applicability to regional languages and usability in resource-constrained settings. This study utilized three datasets from the Tamil, Devanagari and Telugu languages to demonstrate the exceptional performance of the NQT classifier in character recognition through the minimization of misclassification errors. The classifier exhibited consistently low misclassification error rates across a diverse range of characters, outperforming existing deep learning models. Notably, the NQT classifier achieved an overall accuracy, precision, recall and F1-measure of 96%, 95%, 97% and 96%, respectively, across the three datasets, surpassing the performance of popular transfer learning models. Furthermore, a comprehensive computational efficiency analysis highlighted the superior efficiency of the NQT classifier in terms of training time, CPU and GPU utilization and memory requirements, underscoring its potential for practical applications in resource-constrained environments.

ARTICLE HISTORY

Received 11 February 2024
Revised 4 April 2024
Accepted 18 April 2024

KEYWORDS

machine learning; deep learning; classification; variable length; nearest class; speeded-up robust features; handwritten character recognition

REVIEWING EDITOR

Swadesh Kumar Singh,
Gokaraju Rangaraju Institute
of Engineering and
Technology, India

SUBJECTS



Artificial Intelligence;
Automation; Computational
Linguistic & Language
Recognition

1. Introduction

Handwritten character recognition (HCR) systems play a pivotal role in various applications, particularly in countries such as India, where multiple languages and scripts coexist (Kumar et al., 2011). As a nation with rich linguistic and cultural tapestry, the development of efficient HCR systems is paramount (Pal et al., 2012). Such systems have applications in document digitization, language preservation and the automation of administrative tasks (Bappi & Abu, 2024). Despite advancements in technology,

recognizing handwritten characters poses significant challenges, primarily owing to the natural variability in writing styles, shapes and regional linguistic nuances (Alom et al., 2018; AIKendi et al., 2024).

In recent years, the field of HCR systems has witnessed significant advancements propelled by the integration of deep-learning models (Weng & Xia, 2019; Saqib et al., 2022; Wang et al., 2023). These models, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have demonstrated remarkable success in extracting intricate features and patterns from handwritten images,

CONTACT Ashlin Deepa R. N.  rndeepa.pradeep@gmail.com  Department of Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, India

© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

leading to enhanced recognition accuracies (Wu et al., 2014). Deep learning models learn hierarchical representations, capture complex relationships within the data and adapt to nuanced variations in handwritten characters (Taye, 2023). However, despite their advantages, these models exhibit certain disadvantages, particularly in the context of HCR. One notable challenge is the demand for substantial labeled datasets for effective training, which often requires extensive manual annotation. Moreover, deep learning models may struggle with generalization when exposed to diverse handwriting styles or languages, hindering their adaptability in multilingual and culturally varied settings (Vinjit et al., 2020; Memon et al., 2020). Additionally, deep learning models, especially those with intricate architectures, often require substantial computational resources for both training and inference, contributing to high computational burden (Chen et al., 2020; Li et al., 2023). This demand for computational power can limit the accessibility of these models to resource-constrained environments (Habib & Qureshi, 2020; Lawrence & Zhang, 2019; Lamrini et al., 2023). Furthermore, the interpretability of deep learning models remains a concern, as complex internal representations may make it challenging to understand and interpret the decision-making process (Meng et al., 2022; Sheikhalishahi et al., 2023).

This study proposes the development of an efficient and lightweight HCR system specifically designed for Indian native languages, including Tamil and Telugu scripts. The aim of this system is to operate effectively on devices with limited computational resources. This research addresses two key challenges inherent in HCR: significant variability in character shapes and the generation of variable-length feature vectors during extraction. To address these challenges, we introduced the nearest quad-tree (NQT) classifier. This novel approach combines local feature-level decisions with a global decision-making process to improve recognition accuracy.

This research introduces the NQT classifier, a reliable method for recognizing handwritten characters in Indian languages, such as Tamil and Telugu. These languages pose unique challenges owing to their varied writing styles, shapes and linguistic features, which can hinder traditional recognition techniques. The NQT classifier overcomes these difficulties by using a quad-tree structure that analyzes the spatial relationships between insert points (IPs) within the characters. Furthermore, the NQT classifier stands out for its computational efficiency compared to deep learning models. Although deep

learning models often require significant computing power, the NQT classifier operates efficiently, making it ideal for use in environments with limited resources. This efficiency enhances accessibility and practicality, ensuring that the recognition process remains fast and accurate, even for devices with limited computing capabilities. This research highlights the NQT classifier as a promising solution that balances accuracy and computational efficiency, making it a valuable tool for HCR in languages with complex and diverse writing styles.

This research offers several significant contributions to the field of HCR, particularly for native Indian languages.

This research makes several significant contributions to the field of HCR, particularly tailored to Indian native languages, such as Tamil and Telugu scripts. The key contributions of this study include the following:

1. **NQT classifier:** Introduction of the NQT classifier is a pivotal contribution. This classifier is specifically designed to handle the complexities and variations inherent in the writing styles, shapes and linguistic nuances present in Indian scripts.
2. **Local and global decision-making:** This study proposes a novel approach that combines local feature-level decisions with a global decision making process. This dual decision-making strategy enhanced the accuracy of HCR. The classifier not only focuses on individual features but also considers their collective impact, leading to more reliable and context-aware recognition.
3. **Efficiency and lightweight design:** The NQT classifier is notable for its computational efficiency. Unlike deep learning models, which often require extensive computational resources, the NQT classifier is designed to operate effectively on devices with limited resources.
4. **Applicability to regional languages:** A noteworthy contribution of the proposed method is its adaptability to regional languages, with a specific focus on Tamil and Telugu scripts.

These key contributions underline the significance of this research and its potential to advance the field of HCR, particularly for the under-resourced Tamil and other regional languages with similar characteristics.

2. Literature review

Kavitha and Srimathi (2019) proposed a method for Handwritten Tamil Character Recognition using a

CNN. In the pre-processing step, the dataset's 82,929 images were normalized to 64×64 using bilinear interpolation and scaled to a 0, 1 range. Two sets of inputs, one with the original images and the other with inverted images, were used for training. The architecture includes five convolutional layers, two max-pooling layers and two fully connected layers, utilizing ReLU as the activation function. The model employed Xavier initialization for weights and the Adam optimizer with a learning rate of 0.001.

Kowsalya and Periasamy (2019) introduced a novel approach for recognizing Tamil handwritten characters using a modified artificial neural network (ANN) optimized by the Elephant Herding Optimization technique. The method comprises four main processes: preprocessing, segmentation, feature extraction and recognition. Preprocessing involves Gaussian filtering, binarization and skew detection. Segmentation includes line and character segmentations. The feature extraction step was performed on the segmented output.

Pragathi et al. (2019) proposed a character recognition system for handwritten Tamil characters using the VGG 16 approach, a CNN. The study addresses the challenges in recognizing Tamil characters due to a larger category set and similarities between characters. The authors collected a dataset of 156 Tamil characters from Kaggle, each with 100 samples. The VGG 16 algorithm was employed, resizing the images to 224×224 for feature extraction. This study introduces the VGG 16 architecture and emphasizes its effectiveness in baseline feature extraction.

Parihar et al. (2021) proposed a multilingual HCR system using a fine-tuned CNN. The authors highlighted the challenges in HCR due to variations in writing styles across different languages. The CNN architecture was chosen for its efficacy in image classification, utilizing convolution layers and activation functions such as ReLU, max-pooling, dropout and softmax layers.

Moudgil et al. (2023) introduced a Capsule Network (CapsNet) methodology for recognizing Devanagari characters in ancient studies. The study followed a comprehensive methodology, including data collection from two Indian regions, image annotation and segmentation using the Vott tool and CapsNet model implementation. The architecture involves convolution and primary capsules, employing dynamic routing to understand the spatial relationships.

Pal et al. (2008) presented a quadratic classifier-based approach for recognizing offline handwritten

characters of three popular South Indian scripts: Kannada, Telugu and Tamil. Notably, this study addresses the limited research on handwritten Indian characters. The proposed system utilizes directional information for feature extraction computed within the segmented blocks of a character's bounding box. Down-sampling with a Gaussian filter was applied to these directional features, followed by recognition using a modified quadratic discriminant function (MQDF). James et al. (2018) introduced a novel approach to Malay HCR by employing AlexNet-based architecture. Given the linguistic complexity of Malayalam characters, the proposed CNN model utilizes AlexNet for feature extraction, followed by classification using a support vector machine (SVM). A significant contribution of this study is the creation of a Malayalam character dataset that addresses the scarcity of standardized datasets for the language. The augmentation and pre-processing steps enhance the dataset to approximately 180,000 characters.

Poornima Devi and Sornam (2020) proposed a method to classify Tamil characters from palm leaf manuscripts using an ANN. This approach involves contour-based convex-hull bounding-box segmentation for character extraction. The extracted characters are transformed into binary coded values and gray-level co-occurrence matrix (GLCM) features. The modified adaptive backpropagation network (MABPN) algorithm with the Shannon activation function and Nguyen-Widrow weight initialization was used for training. This study explores two input forms: Binary Coded Value and GLCM features, with GLCM showing promising results. Character segmentation involves contour detection owing to its ability to handle curves in Tamil characters. Jangid and Srivastava (2018) introduced a technique for handwritten Devanagari character recognition using deep CNNs (DCNNs). The authors experimented with six DCNN architectures and evaluated their performance on ISIDCHAR and V2DMDCHAR databases. They also explored the use of six adaptive-gradient methods. The DCNN architecture consists of convolutional and fully connected layers, allowing the model to learn features from the color intensity distributions. A layer-wise training approach was proposed, starting with one pair of convolutional and pooling layers and gradually adding more layers. Different adaptive gradient methods such as Adagrad, Adadelta, RMSProp and Adam are discussed.

Deore and Pravin (2020) proposed a two-stage approach for a Devanagari Handwritten Character Recognition System (DHCRS) using a VGG16 deep learning model. In the first stage, a new dataset was generated and a modified VGG16 architecture was

employed. Real-time data augmentation techniques, including translation, rotation, scaling and zooming, have been applied to improve the generalizability. The authors utilized adaptive gradient optimizers, RMSprop and Adam.

The Quadtree-based zoning has been good in the area of OCR. A multi-scale quadtree-based method along with CNN was used for popular Indic scripts (Sarkhel et al., 2017). A voting mechanism through softmax probabilities of CNN was implemented for recognition. Though the recognition accuracy was good, the complexities due to deep learning methods are dominant in this model. Quadtree-based feature extraction along with KNN classification in (Padma and Pasha, 2014) gave 85% accuracy for a dataset of 1200 samples. In another approach (Singh et al., 2020), Quadtree based multi script recognition at block-level has been proposed on 110 pages. The division of images using quadtree approach was proved to be efficient in above methods due to the fact that it divides the region of image based on Centre of Gravity (CG) of white pixels in the image (explained in Section 3.3).

Deep learning techniques are popular in various applications (Liang & Ke, 2023). Existing literature on HCR in Indian regional languages reveals several research gaps. First, current methodologies rely heavily on complex CNN architectures; however, there is a lack of explicit consideration of the computational efficiency of these models, particularly in resource-constrained environments. This oversight limits the practical applicability of these systems in diverse technological landscapes. Additionally, none of the existing classifiers adequately addresses the challenges related to shape variability and variable-length feature vectors in handwritten characters. These gaps emphasize the need for a novel approach such as the proposed NQT classifier, which

aims to combine efficiency with effective recognition, addressing both computational constraints and inherent variability in handwritten characters.

3. Proposed methodology

The proposed methodology for HCR involves several key steps, as illustrated in the process flow diagram in Figure 1. The proposed methodology focuses on accurately recognizing handwritten characters by establishing meaningful connections between the interest points (IPs) in the test and training images. Using Speeded-Up Robust Features (SURF) descriptors (Bay et al., 2008), the Matching Pair Formation Module calculates the minimum distances between IPs, forming matching pairs that serve as the foundation for subsequent analyses. The evaluation of the distances between IPs is crucial in establishing connections, where a straightforward trigonometric method is employed. The NQT classifier enhances the accuracy by utilizing a quad-tree structure based on the center of gravity (CG), introducing the concept of nearest quad-tree-based IPs (NQTIP). This ensures precise classification by considering the spatial distribution of the IPs in the same quadrant. Local decisions are made by calculating the object similarity scores, ranking training images and contributing to the global decision-making process. This methodology addresses challenges in HCR, providing a robust framework for the accurate classification of natural variability and misalignments. The proposed research aims to enhance the effectiveness of HCR systems using a systematic and reliable approach.

3.1. Matching pair formation between IPs

In the process of forming matching pairs between IPs, the proposed method focuses on establishing

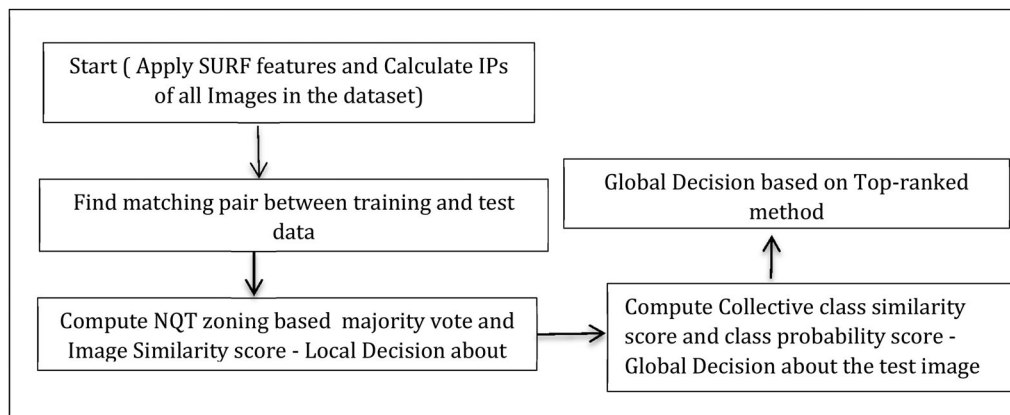


Figure 1. Process flow diagram of proposed handwritten character recognition.

meaningful connections between IPs from the test and training images (Ashlin Deepa & Rajeswara Rao, 2020). The Matching Pair Formation Module utilizes SURF descriptors extracted from the images. For a given test image and its corresponding training image, the module calculates the minimum distance d_i between the SURF descriptors of IPs lnZ_i in the test image and $lnijc$ in the training image. Matching pairs are then formed based on the minimum distance criterion, ensuring that IPs with the closest SURF descriptors are paired. Mathematically, the matching pair formation can be expressed as

$$\text{Matching Pair}_{ij} = \{(lnZ_i, lnijc) \mid d_i = \min(\text{dist}(lnZ_i, lnijc))\} \quad (1)$$

This process ensures that each IP in the test image is paired with its closest counterpart in the training image, thereby establishing the basis for subsequent quadrant-based analyses and classification. The resulting matching pairs serve as a foundation for the NQT-based zoning classifier, contributing to the effective comparison and recognition of handwritten characters. Table 1 provides detailed information on the IPs of the test image, IPs of the training images, their matching pairs, quadrant information with border details and the calculation of image similarity scores. As shown in Table 1, the IPs of the test image Z, denoted as IPZ1, IPZ2, IPZ3, IPZ4 and IPZ5, were matched with the corresponding IPs in the training images based on the minimum distance criterion. Table 2 provides additional insights into the determination of class Z based on global decision-making. Collective class similarity scores (Sg(c)) were computed for each class and the class probability scores for each rank (Sp) were determined. These scores play a crucial role in the recognition process, where the matching pairs formed between the IPs in the test and training images contribute to the effective comparison and recognition of handwritten characters. Figure 2(a) illustrates the matching pair formation process between the IPs of D11 and Z, emphasizing that the solid line indicates the best match with the minimum distance. Figure 2(b) shows the best matches between the IPs of D11 and Z, providing a visual representation of the key steps in determining the matching pairs based on the SURF descriptors.

3.2. Evaluating the distance between IPs

In the context of the proposed research, the evaluation of distances between IPs is a critical step in establishing meaningful connections between

images with different class labels (Ashlin et al., 2023). The IPs are organized into sets, denoted as V for images with class labels d and U for images with a distinct label \bar{d} , where d and \bar{d} belong to the Class Label. Each set comprises IPs represented as IP_{ab_d} and $IP_{pq_{\bar{d}}}$, where a , b , p and q range from one to the respective number of IPs in the images. A straightforward trigonometric method was employed to calculate the distance between the IPs. The distance (d) between the two IPs, represented as IP_{ab_d} and $IP_{pq_{\bar{d}}}$, is computed using Equation (2):

$$d = \frac{|IP_{ab_d} - IP_{pq_{\bar{d}}}|}{\sqrt{2}} \quad (2)$$

Subsequently, the minimum distance (d_i) is determined by repeating this calculation for every IP in U and as expressed by Equation (3):

$$d_i = \min \left\{ \frac{|IP_{ab_d} - IP_{pq_{\bar{d}}}|}{\sqrt{2}} \right\} \quad (3)$$

Here, p varies from 1 to the total number of IPs in the image q . The IP_{ab_d} with the minimum distance d_i forms a matching pair with the corresponding $IP_{pq_{\bar{d}}}$, signifying that the a th IP of image b from class d has the closest resemblance to the p th IP of image q from class \bar{d} . This process is iteratively applied to all IPs in set A , leading to the determination of matching pairs of IPs for image b , as illustrated in Figure 1. This methodology ensures the establishment of significant connections between IPs, laying the foundation for subsequent analyses and classifications in the research framework.

3.3. Computing Nearest quad-tree (NQT) based zoning from SURF descriptors

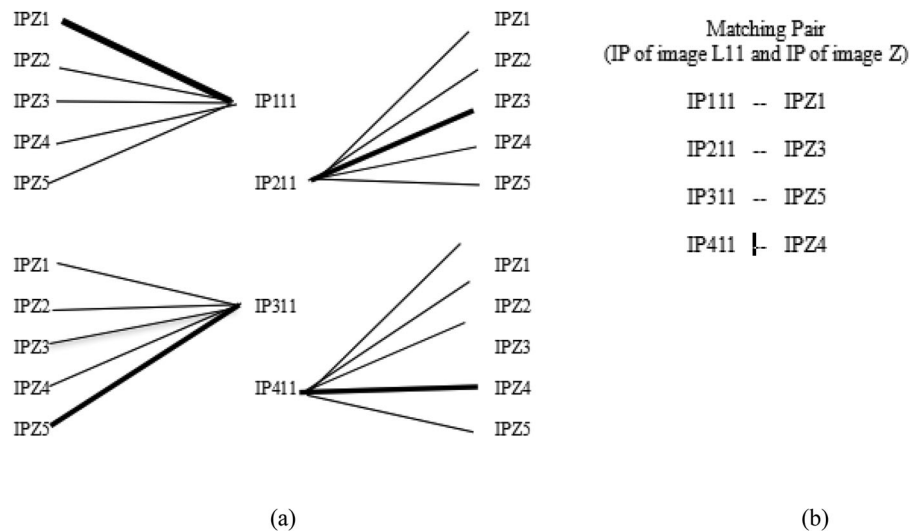
Quad-tree-based zoning methods offer enhanced information compared with fixed-size zoning methods. This approach recursively subdividing a two-dimensional space into four regions. The partitioning of a character image is achieved by drawing vertical and horizontal lines through the CG of the white pixels in the region. The total number of subregions in a quad-tree structure is 4^d , where d represents the depth of the quad-tree structure. The CG coordinates of any image frame (C_x, C_y) were calculated using the following formulas:

$$C_x = \frac{1}{mn} \sum_{mn} x \cdot f(x, y) \quad (3)$$

$$C_y = \frac{1}{mn} \sum_{mn} y \cdot f(x, y) \quad (4)$$

Table 2. Determination of the class of Z based on global decision. Z is classified to the class D2.

Class, c	Collective class similarity scores, $S_g(c)$			Class probability score for each rank S_p		
	Rank			Rank		
	1	2	3	1	2	3
D1	0.5	0.9	1.15	$0.5/0.75 = 0.71$	$0.9/1.35 = 0.66$	$1.15/1.39 = 0.82$
D2	0.75	1.35	1.39	$0.75/0.75 = 1$	$1.35/1.35 = 1$	$1.39/1.39 = 1$
D3	.3	.5	–	$0.3/0.75 = 0.4$	$0.5/1.35 = 0.37$	

**Figure 2.** (a). Matching pair formation between the IPs of D11 and Z. Solid line indicates the best match with minimum distance between IPs. (b): Best matches between IPs of D11 and Z.

Here, $f(x,y)$ takes a value of one for all white pixels and zero for all dark pixels. The CG-based quad-tree partitioning illustrated in Figure 3 demonstrates its advantages over equal partitioning. In our implementation, a depth of one was used.

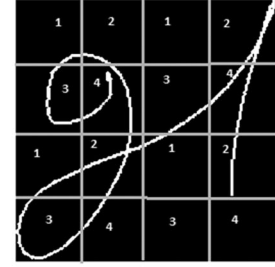
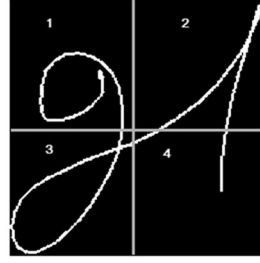
To facilitate the comparison of two images, SURF descriptors were generated and matching pairs were formed based on the minimum distance, as per formula (3). The IPs from the matching pairs correspond to the physical structures of images with the highest similarity (Ashlin Deepa & Rajeswara Rao, 2020). However, in some cases, local structure resemblance leads to incorrect pairings, where an IP in one region is incorrectly matched with an IP in a completely different region. Existing classifiers, such as NIP and NA, which rely on standard deviation and angle thresholds, do not detect such errors. To address this, a novel classifier, NQT-based zoning, is developed. The NQT classifier, based on the region in which an IP is located, provides a solution for accurately classifying the IPs. This introduces an advantage, especially in cases where a correct IP might be misclassified by previous approaches, such as NIP and NA. The NQT classifier considers the regional context, allowing for a more accurate classification of IPs based on their spatial location.

3.3.1. Determination of nearest quad-tree based interest point (NQTIP)

The NQTIP, referred to as NF, plays a crucial role in ensuring the accurate classification of test images based on the spatial distribution of IPs in the images. Image classification at the local level relies on the comparison of images based on character regions. The inherent variability in handwritten characters, including differences in shape and size between individuals, poses a challenge for automatic character recognition. To address this challenge, pre-processing steps, such as determining the bounding box of all images before extracting SURF, are applied to reduce localization errors.

The method employs a quad-tree structure that utilizes the CG for zoning images, thereby significantly minimizing misalignment problems. The methodology adhered to Definitions 1 and 2 for the determination of NQTIP.

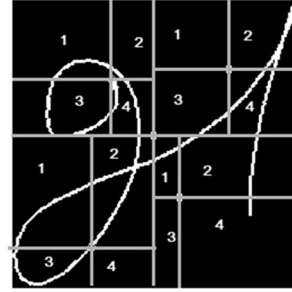
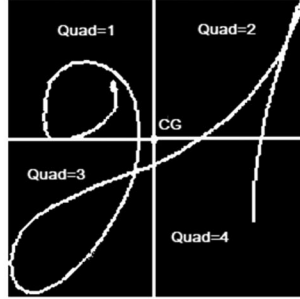
Definition 1: An IP i , belonging to a set of 'M' IPs ($i = 1, \dots, M$) of a training image j , is classified as one of the NQTIP of the given training image if 'k' is the IP of the test image and there exists a matching pair (i,k) , where 'i' and 'k' are present in the same quadrants of training and testing images, respectively.



(a) Sample image Ah

(b) Equal partitioning with depth 1

(c) Equal partitioning with depth 2



(d) CG based quad-tree partitioning with depth 1

(e) CG based quad-tree partitioning with depth 2

Figure 3. Equal partitioning vs. quad-tree partitioning. (a) Sample image Ah (b) Equal partitioning with depth 1 (c) Equal partitioning with depth 2. (d) CG-based quad-tree partitioning with depth 1 (e) CG-based quad-tree partitioning with depth 2.

Definition 2: If an IP i belongs to quadrant ' Q_m ' in an image ' I_i ', and IP k belongs to quadrant ' Q_n ' in an image ' I_k ', and IPs ' i ' & ' k ' form a matching pair (i, k) , then ' $i|k$ ' is classified as NQTIP if and only if ' $m = n$ ' where ' m ' where and ' m ' and ' n ' are quadrant numbers, and ' I_i ' and ' I_k ' and ' I_k ' are training and testing images, or vice versa.

The criteria from Definitions 1 and 2 state that the IP of a training image in a matching pair with the IP of the test image is an NQTIP when both IPs belong to the same quadrant of the training and test images. The concept of NQTIP is illustrated through a set of misclassifications in Figure 4(a–c).

In these figures, the colored lines represent the connections between the IPs of a matching pair in the images used for comparison. It is observed that the endpoints of the lines representing IPs belong to different quadrants in both Figures 4(a,b), resulting in misclassifications. The quad-tree structure detects these misclassifications through the position of quadrants during the comparison between inter-class images, leading to an increased number of IPs that are non-NQTIPs. Figure 5 demonstrates IPs from the same quadrants paired between the test and training images, resulting in a large number of IPs being NQTIP and contributing to the correct classification.

3.3.2. Local NQTIP-based decision using NQT classifier

The NF, called NQTIP, is based on the region in which the IPs of the matching pair from the test and training images are present. Classification under the NQT method focuses on the number of IPs present in the same quadrants of both images used for comparison. Table 1 shows the IPs of the test image Z , IPs of the training images, their matching pairs, quadrants, and the calculation of the 'object similarity score' for each of the training images using NQTIP. The vote for the object similarity score can be calculated as

$$v_i(NQTIP) = \begin{cases} 1, & \text{if } Q_m = Q_n \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where Q_m is the quadrant of training image i and Q_n is the quadrant of the test image Z being compared. Table 6 shows the local similarity score at Level 3 and the local-level decision using the NQT classifier. The top-ranked training object is D12 (rank 1) with a similarity score of 0.75, which belongs to class D2. Hence, the test image is classified as class D2. The steps of the NQTIP classifier are presented in Algorithm 1.

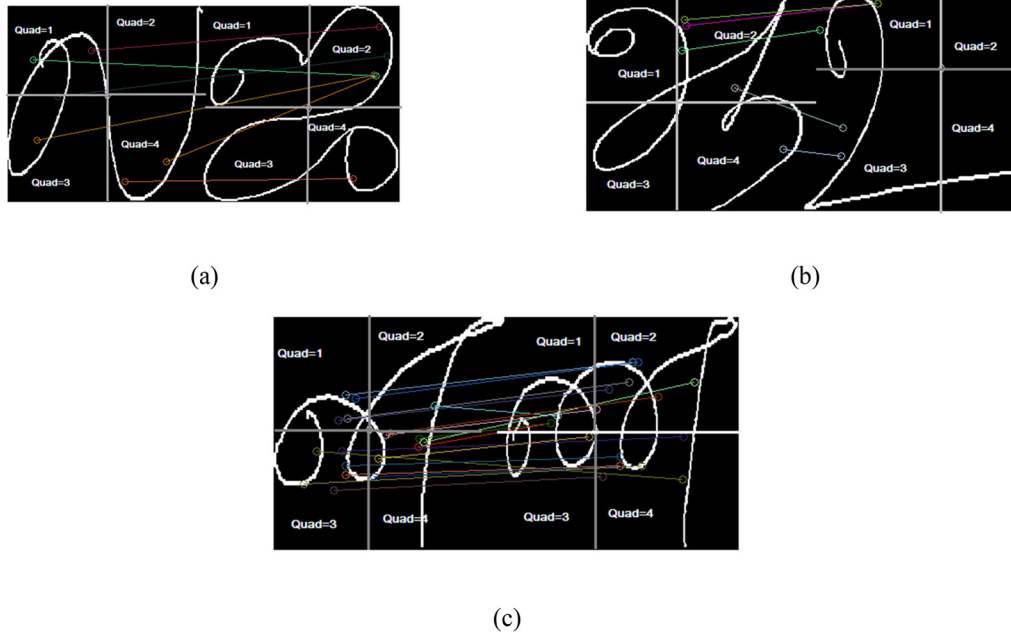


Figure 4. Intra-class images displaying incorrect matching are depicted in Figure (a). In this scenario, insertion points (IPs) from the first and third quadrants of the initial image 'la' erroneously pair with the second quadrant of the second image 'ja'. These pairings result in $nas\ vote = 1$. Figure (b) illustrates another instance where three IPs from quadrant 2 of the training image 'aah' mistakenly match with IPs of the letter 'uh' in quadrant 1. Moreover, an IP from quadrant 2 and another from quadrant 4 of 'aah' erroneously pair with quadrant 3 of image 'uh,' also yielding $nas\ vote = 1$. Moving to Figure (c), we examine inter-class Tamil characters 'na' and 'nna' for comparison. Despite the majority of lines connecting IPs falling within the upper and lower boundaries of the na threshold, there are a few exceptions. Notably, the misclassification of IPs in all three cases generates angles within the upper and lower bounds of the na classifier, resulting in positive votes $vi = 1$. However, by considering the quadrants in which the IPs of the matching pair are present, we can rectify these misclassifications.

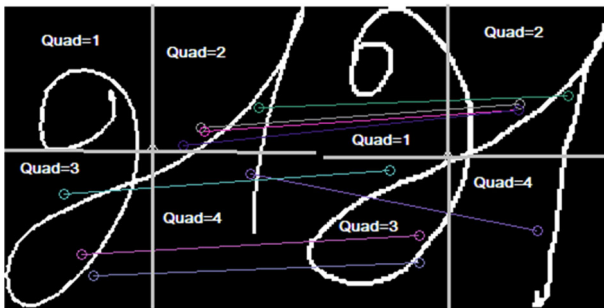


Figure 5. Two inter-class Tamil character images of class 'Ah' having most of the IPs in the matching pairs in the same quadrants.

Algorithm 1: NQT classifier

Following are the steps in the proposed NIP-QT classifier:

1. Generate a bounding box for each training image D_{ij} and resize it to 256×256 . Extract SURF descriptors from training images to generate IPs such as In_{ijc} .
2. Generate a bounding box for the test image Z and resize it to 256×256 . Extract SURF descriptors from the test image to generate IPs such as IPZ_i .
3. Calculate the CG of the test and training images, and generate quadrants with a depth of 1.
4. For each training image D_{ij} , do the following.
 - a. Calculate the IPs matching pair (IPZ_i, In_{ijc}) between the IPs of test image Z and training image D_{ij} after finding the minimum distance d_i using (2). For each matching pair, perform (b) and (c).
 - b. Find quadrant Q_m of In_{ijc} in training image D_{ij} . Find quadrant Q_n of IPZ_i in the test image.
 - c. For each IP of D_{ij} , determine NQTIP based on Definitions 1 and 2 and calculate vote $vi(NQTIP)$ using (3).
5. Image similarity scores for training image D_{ij} are calculated by counting the total number of votes $vi = 1$ associated with that image as shown in Table 1.
6. The training images were ranked as Rank 1, Rank 2, and Rank 3 based on the image similarity values, as shown in Table 1.
7. Image similarity scores of 'm' top-ranked images are taken to obtain local decision using NQTIPs. The classes of the training images having the highest scores at top ranks (Rank 1, Rank 2, and

Rank 3) are maintained for the global-level decision as shown in Table 2.

When inter-class images are compared, an increased number of non-NQTIPs are detected, whereas in intra-class image comparison, the number of NQTIPs is large. Figure 3(a–c) shows a quadrant-based inter-class image comparison. Only a few IP-matching pairs with connecting lines are shown in the figures. We can see that the angle between the IPs in the matching pairs is within the allowed upper- and lower-bound limits. Therefore, as per the NA’s classifier, these points are considered for our earlier approach NA’s to produce a positive result during similarity voting ($v_i = 1$). However, the NQT classifier detects dissimilarity with respect to the region and decreases the inter-class similarity. Figures 5 and 6 show intra-class images for comparison, where we can see that most of the IPs in the matching pair belong to the same quadrants ($v_i = 1$) in the respective images. Only two lines connecting IPs A and B and C and D are present in different quadrants, as shown in Figure 6. A modification to NQTIP was performed to catch missed IPs in the same quadrants because of their existence at the neighboring quadrant border.

4. Results and discussion

This section presents the results and discussion of the proposed NQT classifier for HCR. It covers the system and parameter settings used, and provides an in-depth analysis of the recognition accuracy across different datasets, including Tamil, Devanagari and Telugu. The discussion further explores how the NQT classifier minimizes misclassification, offering insights into its effectiveness in reducing errors related to writing style. In addition, it addresses the

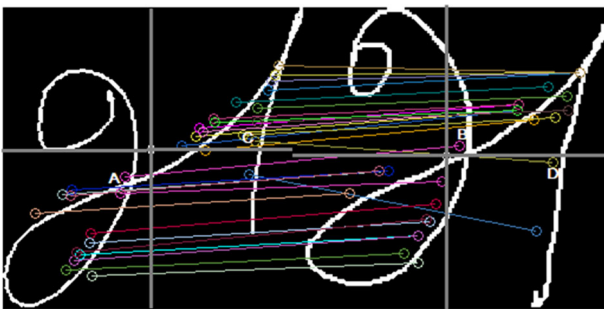


Figure 6. Two intra-class images of Tamil character ‘Ah’ with most of the IPs are inside the same quadrants of respective images showing highest similarity.

computational efficiency of the NQT classifier, highlighting its practicality as an alternative to deep learning models, especially in situations with limited resources.

4.1. System settings

The system setup for this research involves utilizing a ThinkPad E16 laptop with a 13th Gen Intel i5 processor running on Windows 11. The laptop was equipped with Intel Iris Xe Graphics and 16GB of DDR4 RAM. The software components included MATLAB 2020 supplemented with OpenCV and MATLAB deep learning libraries. The hyperparameters employed in the system configuration were set as follows: learning rate of 0.001, batch size of 32, training for 200 epochs, utilizing the Adam optimizer, and employing Categorical Cross-entropy as the loss function for effective model training and evaluation.

4.2. Data collection

This research used three datasets of handwritten characters from different Indian languages: Tamil, Devanagari and Telugu. These datasets were created specifically for training and testing the handwriting recognition systems.

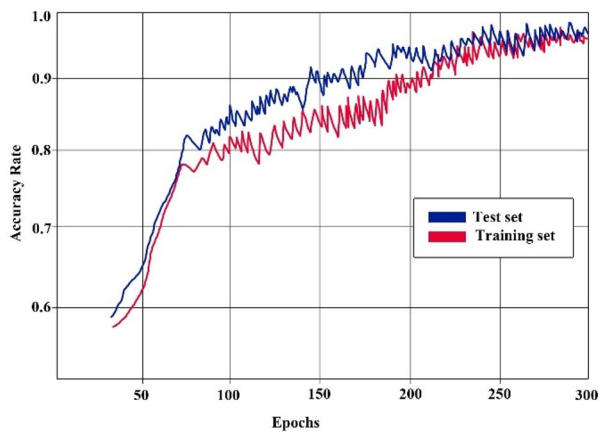
- **Tamil Hp Dataset:** This dataset contains approximately 7500 isolated Tamil characters written by various people including students, adults, and professionals. The characters were collected using special pen tablets that captured strokes and movements as people wrote (Isolated Handwritten Tamil Character Dataset, 2006).
- **Devanagari Dataset:** This dataset includes over 15,500 handwritten Devanagari characters used in the Hindi language. Similar to the Tamil dataset, the dataset was collected using pen tablets from a diverse group of writers.
- **Telugu Handwritten Character Dataset:** This dataset consists of approximately 10,800 isolated Telugu characters. The data were gathered using the same method as for the other two datasets, with people writing on pen tablets.

The dataset was divided, with 80% of the randomly selected data allocated for training and the remaining 20% for testing.

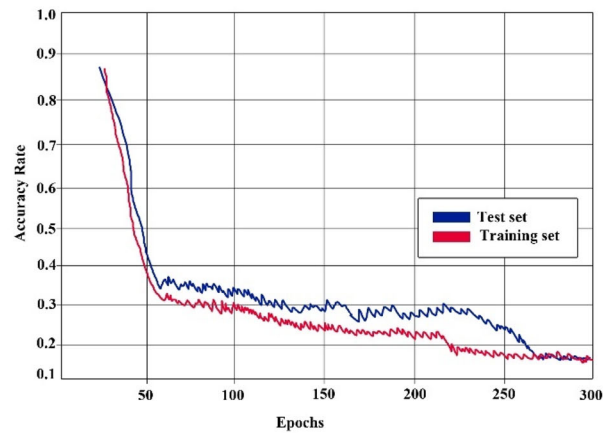
4.3. Accuracy and loss comparison: training vs. testing data

We assessed the training efficiency of the proposed NQT classifier by closely examining its training and

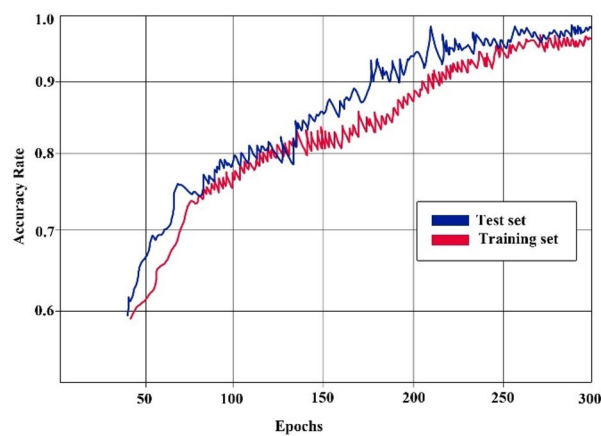
testing accuracy as well as its training and testing loss across three different datasets: Tamil, Telugu, and Devanagari. Figure 7 visually compares the training versus testing accuracy and training versus



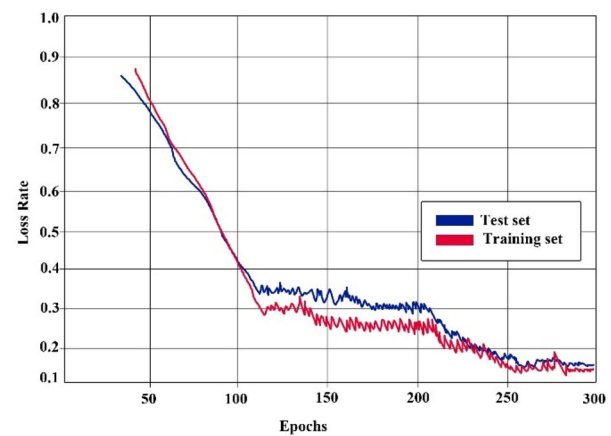
(a)



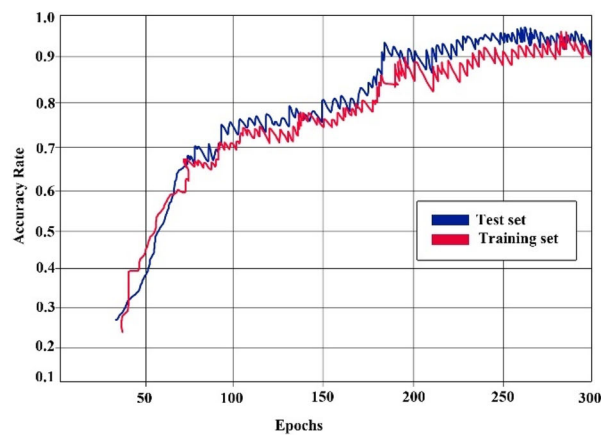
(b)



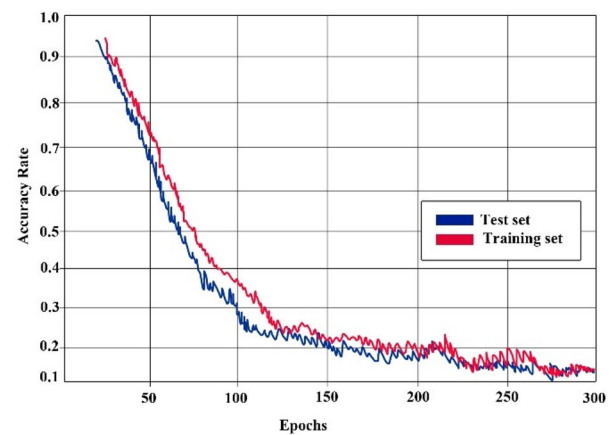
(c)



(d)



(e)



(f)

Figure 7. (a) Proposed model training vs. testing accuracy for Tamil dataset. (b) Proposed model training vs. testing loss for Tamil dataset. (c) Proposed model training vs. testing accuracy for Telugu dataset. (d) Proposed model training vs. testing accuracy for Telugu dataset. (e) Proposed model training vs. testing accuracy for Devanagari dataset. (f) Proposed model training vs. testing accuracy for Devanagari dataset.

testing loss for each dataset, providing insights into the efficiency and generalizability of the proposed method.

Tamil Dataset: Figure 7(a) shows the training and testing accuracy of the model on the Tamil dataset. The plot reveals that the training accuracy steadily increases as the model goes through training cycles (epochs), indicating its ability to learn from the training data. At the same time, the testing accuracy remained slightly higher, suggesting good generalization to the unseen data. This small difference in testing accuracy implies that the proposed method effectively avoids overfitting, ensuring that the model performs well not only on the training set, but also on new, unseen data. **Telugu and Devanagari Datasets:** Similar trends were observed for the Telugu and Devanagari datasets. Figure 7(c,e) shows steady increases in the training accuracy, reflecting the model's ability to learn the underlying patterns of the characters. In both cases, the testing accuracy consistently remained slightly higher, highlighting the classifier's ability to generalize effectively. Figure 7(d,f) depicts the training and testing losses for these datasets, revealing patterns similar to the Tamil dataset. The training loss decreases as the model learns from the data, whereas the testing loss remains consistently lower, indicating robust generalization.

The detailed training efficiency analysis reveals that the proposed NQT classifier effectively learns from the training data, minimizes training loss, and generalizes well to the testing data across all three datasets. The small differences between the training and testing metrics suggest that the model avoids overfitting and achieves a robust performance in recognizing handwritten characters in various linguistic contexts.

4.4. Accuracy analysis

The accuracy analysis section delves into a comparative assessment of crucial accuracy metrics for the proposed model and popular transfer learning models, namely VGG-16, LeNet, ResNet, MobileNet, GoogleNet and AlexNet, across three datasets: Tamil, Telugu and Devanagari. Specifically, the evaluation focused on average accuracy metrics for the Tamil, Telugu and Hindi vowels. Key accuracy metrics include Accuracy, Precision, Recall and F1-measure, each contributing to a comprehensive understanding of the performance of the proposed model compared with established deep learning benchmarks. The accuracy metrics were calculated using Equations (6–9).

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{\text{Total samples}} \quad (6)$$

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \quad (7)$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \quad (8)$$

$$\text{F1-measure} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (9)$$

Tables 3–5 provide a detailed comparison of the average classification accuracy for Tamil and Hindi vowels between the proposed method and several existing transfer learning models, such as VGG-16, LeNet, AlexNet, MobileNet and GoogleNet. Each table includes key metrics, such as accuracy, precision, recall and F1-measure. In Table 3, which focuses on Tamil vowels, the proposed model consistently performed better than all other models across all metrics. It achieved higher values in accuracy (0.95), precision (0.94), recall (0.96) and F1-measure (0.95). Table 4, focusing on Hindi vowels, further supports the superiority of the proposed model by showing higher values in all metrics than the other models. Similarly, Table 5 reinforces the effectiveness of the proposed model in maintaining a high

Table 3. Comparison of the average classification accuracy of Tamil vowels between the proposed method and existing transfer learning models.

Models	Accuracy	Precision	Recall	F1-measure
Proposed model	0.95	0.94	0.96	0.95
VGG-16	0.89	0.88	0.90	0.89
LeNet	0.92	0.91	0.93	0.92
AlexNet	0.91	0.90	0.92	0.91
MobileNet	0.94	0.93	0.95	0.94
GoogleNet	0.88	0.87	0.89	0.88

Table 4. Comparison of the average classification accuracy of Hindi vowels between the proposed method and existing transfer learning models.

Models	Accuracy	Precision	Recall	F1-measure
Proposed model	0.98	0.97	0.99	0.98
VGG-16	0.78	0.77	0.79	0.78
LeNet	0.81	0.80	0.82	0.81
AlexNet	0.79	0.78	0.80	0.79
MobileNet	0.84	0.83	0.85	0.84
GoogleNet	0.75	0.74	0.76	0.75

Table 5. Comparison of the average classification accuracy of Tamil vowels between the proposed method and existing transfer learning models.

Models	Accuracy	Precision	Recall	F1-measure
Proposed model	0.95	0.94	0.96	0.95
VGG-16	0.91	0.90	0.92	0.91
LeNet	0.92	0.91	0.93	0.92
AlexNet	0.93	0.92	0.94	0.93
MobileNet	0.89	0.88	0.90	0.89
GoogleNet	0.92	0.91	0.93	0.92

accuracy for Tamil vowels. The proposed method achieved an overall accuracy, precision, recall, and F1-measure of 96%, 95%, 97% and 96%, respectively, for all three datasets.

4.4. Mitigation of misclassification with NQT classifier

An analysis of misclassification with the NQT classifier is conducted in this section, and its performance is compared with existing transfer learning models. The evaluation focuses on the efficiency of misclassification, specifically testing with Tamil vowels in combination with vowel-consonants from the Tamil handwritten dataset. To ensure comprehensive coverage, each Tamil vowel with vowel-consonants was tested across 10 different handwritten styles. Table 6 presents the misclassification results of Tamil vowels with vowel-consonants using various classifiers, including VGG-16, LeNet, AlexNet, MobileNet, GoogleNet and the NQT classifier. Each row corresponds to a specific Tamil vowel, denoting its real shape, whereas the numerical values indicate misclassification occurrences of each classifier. For instance, the Tamil character 'அ' with vowel-consonants had no misclassification by any of the classifiers except LeNet, which made one error. The table's average misclassification rates across all vowel characters are as follows: VGG-16 (1.41), LeNet (1), AlexNet (2.46), MobileNet (1.53), GoogleNet (1.36) and the NQT classifier (0.33). These values provide insights into the classifiers' performance in recognizing Tamil vowels within the context of vowel-consonants, with the NQT classifier exhibiting the lowest average misclassification rate among all models.

4.5. Computational efficiency analysis

In the computational efficiency analysis section, we examine the training time, CPU utilization, GPU utilization and memory utilization of the proposed method and the existing transfer learning methods for the three datasets mentioned above. Figure 8 presents the training-time analysis for different models across the Tamil Hp, Devanagari and Telugu Handwritten Character datasets. The proposed model exhibits notably shorter training times than VGG-16, LeNet, AlexNet and MobileNet, indicating superior efficiency in learning from the datasets. As shown in Figure 9, which illustrates the CPU utilization percentages, the proposed model consistently maintained lower CPU utilization across all datasets. This suggests that the proposed method is less

resource-intensive in the central processing unit compared with the other models considered. Figure 9 extends the analysis to GPU utilization percentages, demonstrating the efficiency of each model in utilizing the graphical processing resources. The proposed model consistently displayed lower GPU utilization percentages, signifying a more economical use of graphical processing capabilities. Finally, Figure 10 focuses on the memory utilization in megabytes. The proposed model consistently requires less memory than VGG-16, LeNet, AlexNet and MobileNet across all datasets, emphasizing its efficiency in handling memory resources. These computational efficiency metrics collectively underscore the advantageous performance of the proposed method in terms of the training time, CPU and GPU utilization and memory requirements (Figure 11).

4.6. Discussion

The primary purpose of this research is to address the challenges in HCR, particularly in the context of diverse Indian languages, such as Tamil and Telugu. The inherent variability in writing styles poses a significant obstacle to achieving accurate recognition, and existing methods, especially those based on deep learning models, encounter limitations in situations with constrained computational resources.

The proposed NQT classifier emerged as a solution to these challenges, offering a novel approach that combines efficient feature extraction with an intelligent zoning mechanism. One of the key problems addressed in this study is the minimization of misclassification errors, which are prevalent in HCR tasks. By utilizing the spatial relationships of IPs through the NQT method, the classifier provides a detailed understanding of the characters, enabling it to distinguish subtle variations in writing style. This is particularly crucial in the context of Indian languages, where characters exhibit intricate details that vary significantly across writers.

The NQT method introduces a unique matching pair formation and distance calculation process, followed by quad-tree partitioning based on the CG. The subsequent classification of IPs within the same quadrant (NQTIP) enhances the system's ability to capture the intricate spatial features of the characters. The results highlight the efficacy of this methodology in achieving a high accuracy across the Tamil, Devanagari and Telugu datasets. The proposed model not only demonstrates robust training

Table 6. Misclassification results of Tamil vowels with vowels-consonants using various classifiers.

Handwritten Tamil character	Real shape	Misclassification results					
		VGG-16	LeNet	AlexNet	MobileNet	GoogleNet	NQT
அ	அ	0	0	1	0	0	0
ஆ	ஆ	1	1	2	0	1	0
இ	இ	0	0	2	1	1	1
ஈ	ஈ	2	2	3	2	1	1
உ	உ	1	0	1	1	2	0
ஊ	ஊ	3	2	4	3	2	1
எ	எ	1	0	2	2	1	0
ஏ	ஏ	1	0	2	1	1	0
ஐ	ஐ	1	1	3	3	1	0
ஒ	ஒ	2	2	2	1	2	0
ஔ	ஔ	2	0	2	1	2	0
ஃ	ஃ	1	3	3	1	1	2
க	க	1	0	3	2	1	0
ங	ங	1	0	1	2	1	0
ச	ச	3	1	4	2	2	0
ஞ	ஞ	2	2	3	1	2	1
ட	ட	1	0	3	3	1	0
ண	ண	1	3	3	1	1	0
த	த	1	0	3	1	1	0
ந	ந	2	0	2	1	2	0
ப	ப	2	1	3	2	2	0
ம	ம	1	2	1	2	1	0
ய	ய	1	0	4	2	1	0
ர	ர	1	3	2	1	1	0
ல	ல	2	0	2	3	2	0

(continued)

Table 6. Continued.

Misclassification results							
Handwritten Tamil character	Real shape	VGG-16	LeNet	AlexNet	MobileNet	GoogleNet	NQT
	உ	2	0	3	1	2	0
	ழ	1	1	2	1	1	2
	ஶ	1	2	2	1	1	0
	ஶ	1	1	3	2	2	0
	ஶ	2	3	3	2	2	2
Average		1.41	1	2.46	1.53	1.36	0.33

Misclassification results in bold.

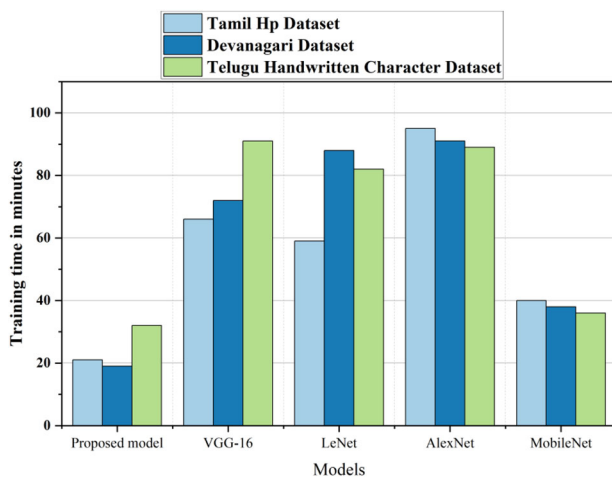


Figure 8. Training time analysis for proposed method with existing transfer learning models.

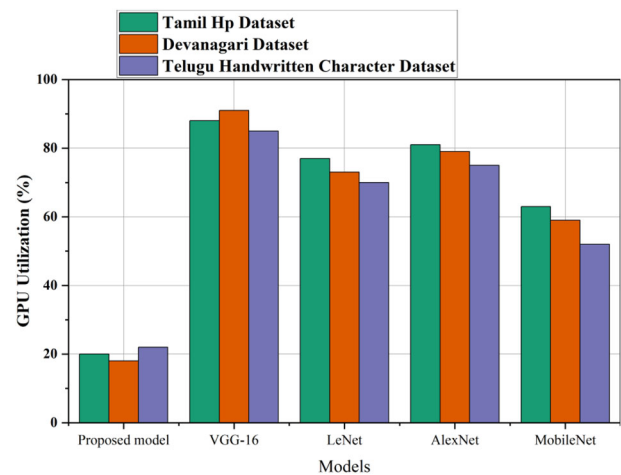


Figure 10. GPU utilization comparison for proposed method with existing transfer learning models.

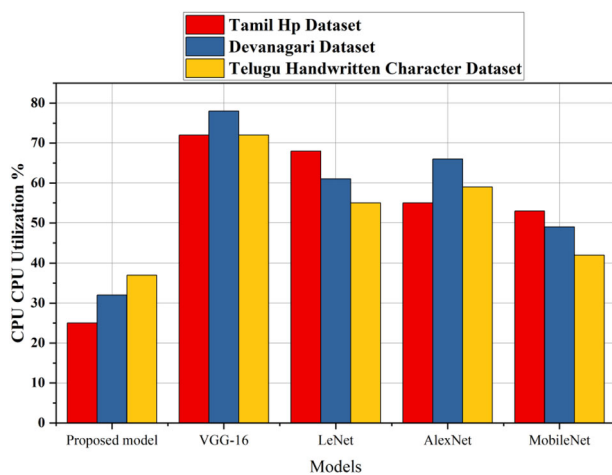


Figure 9. CPU Utilization Comparison for proposed method with existing transfer learning models.

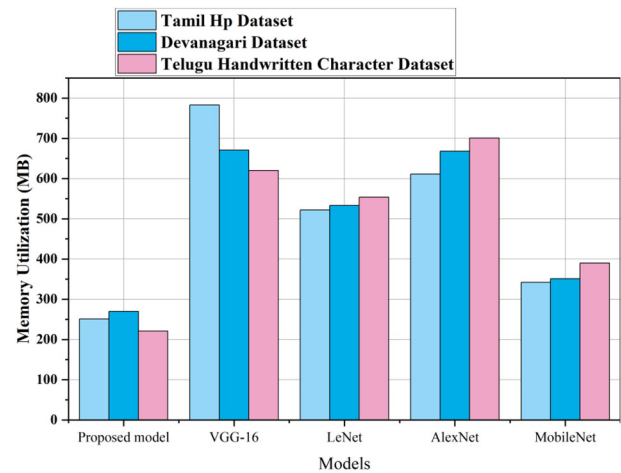


Figure 11. Memory utilization comparison for proposed method with existing transfer learning models.

efficiency, but also shows superior generalization to unseen data, indicating a commendable ability to avoid overfitting.

In comparison with established transfer learning models, such as VGG-16, LeNet, ResNet, MobileNet, GoogleNet and AlexNet, the NQT classifier exhibits

notable improvements in terms of average accuracy, precision, recall and F1-measure. In particular, the misclassification analysis reveals the NQT classifier's superior performance in recognizing Tamil vowels in conjunction with vowel-consonants, showing a remarkably low average misclassification rate compared to its alternatives.

The computational efficiency analysis further emphasizes the practicality of the proposed method. The NQT classifier not only achieves accurate recognition, but also has shorter training times, lower CPU and GPU utilization and reduced memory requirements compared to existing transfer learning models. This efficiency is crucial in scenarios in which computational resources are limited, providing a viable alternative to resource-intensive deep learning models.

Overall, this research successfully addressed the challenges in HCR for Indian languages by introducing the NQT classifier. The proposed methodology excels in minimizing misclassification errors, demonstrates superior recognition accuracy and exhibits commendable computational efficiency. These findings suggest that the NQT classifier is a promising and practical alternative, particularly in situations where resource constraints and varied writing styles pose significant challenges to the existing recognition systems.

5. Conclusion

The primary aim of this study was to address the challenges associated with recognizing subtle variations in writing styles, and the proposed methodology effectively tackles this problem. By utilizing spatial relationships among IPs through the NQT method, the classifier exhibits a refined understanding of characters, enabling it to perceive subtle differences in writing style. The results consistently demonstrate the NQT classifier's ability to minimize misclassification, showing its superior performance compared with established deep learning models. This research significantly contributes to the field by introducing a novel method that not only enhances recognition accuracy, but also offers computational efficiency, making it a practical alternative to resource-intensive deep learning models. The proposed method achieved an overall accuracy, precision, recall and F1-measure of 96%, 95%, 97% and 96%, respectively, for all three datasets. The average misclassification rates further highlighted the superiority of the NQT classifier: VGG-16 (1.41), LeNet (1), AlexNet (2.46), MobileNet (1.53), GoogleNet (1.36) and NQT classifier (0.33). These results underscore the effectiveness of the NQT classifier for handling

complex writing styles. The computational efficiency analysis revealed that the proposed method outperforms existing models in terms of training time, CPU and GPU utilization and memory requirements.

In addition to the achieved milestones, future enhancements could focus on refining the NQT classifier's adaptability to different languages and writing styles, thereby expanding its applicability to a more diverse set of datasets. Further exploration of optimization techniques and fine-tuning parameters may contribute to higher accuracy rates and improved efficiency.

Author contribution

Ashlin Deepa R N and Prasad Chetti conceived and designed the work. P Chandra Sekhar Reddy developed the theory and performed the computation. Sorabh Lakhnpal and Abhishek Joshi drafted the article. Soloveva O.V performed critical revision of the article. All authors discussed the results and contributed to the final manuscript.

Disclosure statement

The author declares no competing interests.

Funding

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

ORCID

Ashlin Deepa R. N.  <http://orcid.org/0000-0002-1742-7516>

Dataset availability statement

HP-Labs-India has established a dataset called hpl-tamil-iso-char for HCR system for Tamil script with 156 character symbols which is available in <https://lipitk.sourceforge.net/datasets/tamilchardata.htm>. The data supporting the findings of this study are available from the corresponding author [Ashlin Deepa R N] on request.

About the authors

Dr. Ashlin Deepa R. N. is an accomplished Associate Professor of Computer Science Engineering at Gokaraju Rangaraju Institute of Engineering and Technology. She earned her doctoral degree in Computer Science Engineering from JNTU Hyderabad, India in 2021. With over 13 years of experience in Additionally, she holds a patent in her field. academia, Dr. Ashlin Deepa R N specializes in areas such as pattern recognition, machine learning, data science, and high-performance computing. Her

extensive research contributions are evidenced by numerous publications in esteemed international journals and conferences. Her enthusiasm for research is further demonstrated by her involvement in various consultancy and research projects. Notably, she serves as the Principal Investigator for a DST-sponsored project sanctioned in 2023, showcasing her leadership and expertise in managing research initiatives.

Dr. Prasad Chetti has established himself as an eminent scholar in the field of Information Technology, earning his Ph.D. from the University of Nebraska-Omaha in 2023. His academic foundation is built on degrees in Computer Science and Engineering from prestigious institutions, including the Indian Institute of Technology Guwahati. Dr. Chetti's teaching repertoire spans a wide range of subjects, from Data Science to Artificial Intelligence, across various esteemed colleges and universities. His research interests, which include Data Science, Artificial Intelligence, and Computational Biology, have led to pivotal contributions in the analysis of civil infrastructures and financial markets using advanced methodologies like population analysis and correlation networks. Dr. Chetti has also mentored numerous student projects, earned accolades for his research, and contributed significantly to academic and university service committees. His dedication to advancing his field is further evidenced by his prolific output of conference proceedings and journal publications, underscoring his commitment to research and education.

Dr. P. Chandra Sekhar Reddy completed his B.Tech in Computer Science & Engineering from G Pulla Reddy Engineering College, Sri Krishna Devaraya University. He received the Master's Degree in M.Tech in Computer Science & Engineering from Jawaharlal Nehru Technological University, Hyderabad. He received his Ph.D Degree in Computer Science & Engineering from Jawaharlal Nehru Technological University Anantapur. He is currently working as a Professor in Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad. He has more than 21 years of teaching experience. His research interests include Image processing, Artificial Intelligence, Data Mining, Networks and Cybersecurity. He has more than 40 publications in various International journals and Conferences. He is also a reviewer and editorial board member for many International journals and International Conferences.

Sorabh Lakhanpal currently serves as Professor, Senior Dean, and Head for the Student Welfare Wing at Lovely Professional University, Punjab, India. His diverse educational journey includes graduation in Indian System of Medicine-Ayurveda, PG in Clinical Microbiology, an MBA in Health Care Services and Ph.D. in Healthcare Management. He holds three granted patents, received grants for various projects, and contributed to national and international journals with scientific publications and presentations.

Prof. (Dr.) Abhishek Joshi is offering his guidance to Uttaranchal University as Executive Director- Students Affairs & IT Services. Prof. (Dr.) Abhishek Joshi pursued his Post Graduation in Political Science and has a Doctorate in Political Science which complements his belief in the creation of a proper system and growth-oriented working

mechanism. Prof. (Dr.) Abhishek Joshi emphasizes on optimum utilization of resources and believes that proper analysis and projection leads to excellence. With an aim to establish Uttaranchal University as a center of excellence, he contributes by steering the University in the right direction through his in-depth analysis and identification of developmental thrust areas. The milestone achieved by the University under his guidance strengthens its faith in the realization of all future aspirations.

Olga Soloveva, associate professor of Kazan State Power Engineering University, Kazan. O.V Soloveva has authored many research publications in international journals.

References

- AlKendi, W., Gechter, F., Heyberger, L., & Guyeux, C. (2024). Advancements and challenges in handwritten text recognition: A comprehensive survey. *Journal of Imaging*, 10(1), 18. <https://doi.org/10.3390/JIMAGING10010018>
- Alom, M. Z., Sidike, P., Hasan, M., Taha, T. M., & Asari, V. K. (2018). Handwritten bangla character recognition using the state-of-the-art deep convolutional neural networks. *Computational Intelligence and Neuroscience*, 2018, 6747098–6747013. <https://doi.org/10.1155/2018/6747098>
- Ashlin Deepa, R. N., & Rajeswara Rao, R. (2020). A novel nearest interest point classifier for offline Tamil handwritten character recognition. *Pattern Analysis and Applications*, 23(1), 199–212. <https://doi.org/10.1007/s10044-018-00776-x>
- Ashlin, R. N., Deepa, S., Narayanan, A., Padthe, & Ramannavar, M. (2023). A reduced feature-set OCR system to recognize handwritten tamil characters using SURF local descriptor. *International Journal of Advanced Computer Science and Applications*, 14(10), 331–344. <https://doi.org/10.14569/IJACSA.2023.0141036>
- Bappi, J. O., & Abu, M. (2024). CBD2023: A hypercomplex bangla handwriting character recognition data for hierarchical class expansion using deep learning. *Data in Brief*, 52, 109909–109909. <https://doi.org/10.1016/j.dib.2023.109909>
- Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3), 346–359. <https://doi.org/10.1016/j.cviu.2007.09.014>
- Chen, C., Zhang, P., Zhang, H., Dai, J., Yi, Y., Zhang, H., & Zhang, Y. (2020). Deep learning on computational-resource-limited platforms: A survey. *Mobile Information Systems*, 2020, 1–19. <https://doi.org/10.1155/2020/8454327>
- Deore, S. P., & Pravin, A. (2020). Devanagari handwritten character recognition using fine-tuned deep convolutional neural network on trivial dataset. *Sādhanā*, 45(1), 2020. <https://doi.org/10.1007/s12046-020-01484-1>
- Habib, G., & Qureshi, S. (2020). Optimization and acceleration of convolutional neural networks: A survey. *Journal of King Saud University - Computer and Information Sciences*, 34(7), 4244–4268. <https://doi.org/10.1016/j.jksuci.2020.10.004>
- Isolated Handwritten Tamil Character Dataset. (2006). <https://lipitk.sourceforge.net/datasets/tamilchardata.htm>, June 2006.
- James, A., Manjusha, J., & Saravanan, C. (2018). Malayalam handwritten character recognition using AlexNet based architecture. *Indonesian Journal of Electrical Engineering*

- and Informatics (IJEEI), 6(4). <https://doi.org/10.11591/ijeei.v6i4.518>
- Jangid, M., & Srivastava, S. (2018). Handwritten devanagari character recognition using layer-wise training of deep convolutional neural networks and adaptive gradient methods. *Journal of Imaging*, 4(2), 41. <https://doi.org/10.3390/jimaging4020041>
- Kavitha, B. R., & Srimathi, C. (2019). Benchmarking on off-line Handwritten Tamil Character Recognition using convolutional neural networks. *Journal of King Saud University - Computer and Information Sciences*.
- Kowsalya, S., & Periasamy, P. S. (2019). Recognition of Tamil handwritten character using modified neural network with aid of elephant herding optimization. *Multimedia Tools and Applications*, 78(17), 25043–25061. <https://doi.org/10.1007/s11042-019-7624-2>
- Kumar, M., Jindal, M. K., & Sharma, R. K. (2011). Review on OCR for handwritten Indian scripts character recognition. *Communications in computer and information science* (pp. 268–276). Springer, Berlin, Heidelberg.
- Lamrini, M., Chkouri, M. Y., & Touhafi, A. (2023). Evaluating the performance of pre-trained convolutional neural network for audio classification on embedded systems for anomaly detection in smart cities. *Sensors*, 23(13), 6227–6227. <https://doi.org/10.3390/s23136227>
- Lawrence, T., & Zhang, L. (2019). IoTNet: An efficient and accurate convolutional neural network for IoT devices. *Sensors*, 19(24), 5541. <https://doi.org/10.3390/s19245541>
- Li, H., Wang, Z., Yue, X., Wang, W., Tomiyama, H., & Meng, L. (2023). An architecture-level analysis on deep learning models for low-impact computations. *Artificial Intelligence Review*, 56(3), 1971–2010. <https://doi.org/10.1007/s10462-022-10221-5>
- Liang, L., & Ke, Y. (2023). User behavior data analysis and product design optimization algorithm based on deep learning. *International Journal on Interactive Design and Manufacturing (IJIDeM)*. <https://doi.org/10.1007/s12008-023-01652-7>
- Memon, J., Sami, M., Khan, R. A., & Uddin, M. (2020). Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR). *IEEE Access*, 8, 142642–142668. <https://doi.org/10.1109/ACCESS.2020.3012542>
- Meng, C., Trinh, L., Xu, N., Enouen, J., & Liu, Y. (2022). Interpretability and fairness evaluation of deep learning models on MIMIC-IV dataset. *Scientific Reports*, 12(1), 7166. <https://doi.org/10.1038/s41598-022-11012-2>
- Moudgil, A., Singh, S., Gautam, V., Rani, S., & Shah, S. H. (2023). Handwritten Devanagari manuscript characters recognition using capsnet. *International Journal of Cognitive Computing in Engineering*, 4, 47–54. <https://doi.org/10.1016/j.ijcce.2023.02.001>
- Padma, M. C., & Pasha, S. (2014). Quadtree based feature extraction technique for recognizing handwritten Kannada characters. *Lecture Notes in Electrical Engineering*, 248:725–735. https://doi.org/10.1007/978-81-322-1157-0_74
- Pal, U., Jayadevan, R., & Sharma, N. (2012). Handwriting recognition in Indian regional scripts. *ACM Transactions on Asian Language Information Processing*, 11(1), 1–35. <https://doi.org/10.1145/2090176.2090177>
- Pal, U., Sharma, N., Wakabayashi, T., & Kimura, F. (2008). *Handwritten character recognition of popular South Indian scripts* (pp. 251–264). Springer eBooks.
- Parihar, G., Rajalakshmi, R., & Bhuvana, J. (2021). *Multi-lingual handwritten character recognition using deep learning* (pp.155–180).
- Poornima Devi, M., & Sornam, M. (2020). Classification of ancient handwritten Tamil characters on palm leaf inscription using modified adaptive backpropagation neural network with GLCM features. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 19(6), 1–24. <https://doi.org/10.1145/3406209>
- Pragathi, M. A., Priyadarshini, K., Saveetha, S., Banu, A. S., & Mohammed Aarif, K. O. (2019). *Handwritten tamil character recognition using deep learning* [Paper presentation]. 2019 International Conference on Vision towards Emerging Trends in Communication and Networking (ViTECoN) (pp. 1–5), Vellore, India. <https://doi.org/10.1109/ViTECoN.2019.8899614>
- Saqib, N., Haque, K. F., Yanambaka, V. P., & Abdelgawad, A. (2022). Convolutional-neural-network-based handwritten character recognition: An approach with massive multi-source data. *Algorithms*, 15(4), 129. <https://doi.org/10.3390/a15040129>
- Sarkhel, R., Das, N., Das, A., Kundu, M., & Nasipuri, M. (2017). A multi-scale deep quad tree based feature extraction method for the recognition of isolated handwritten characters of popular indic scripts. *Pattern Recognition*, 71, 78–93. <https://doi.org/10.1016/j.patcog.2017.05.022>
- Sheikhshah, S., Bhattacharyya, A., Celi, L. A., & Osmani, V. (2023). An interpretable deep learning model for time-series electronic health records: Case study of delirium prediction in critical care. *Artificial Intelligence in Medicine*, 144, 102659. <https://doi.org/10.1016/j.artmed.2023.102659>
- Singh, P. K., Das, S., Sarkar, R., & Nasipuri, M. (2020). *A new approach for texture based script identification at block level using quad tree decomposition*. <https://arxiv.org/abs/2009.07435>.
- Taye, M. M. (2023). Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions. *Computers*, 12(5), 91–91. <https://doi.org/10.3390/computers12050091>
- Vinjit, B. M., Bhojak, M. K., Kumar, S., & Chalak, G. (2020). *A review on handwritten character recognition methods and techniques* [Paper presentation]. 2020 International Conference on Communication and Signal Processing (ICCSPP) (pp. 1224–1228). Chennai, India. <https://doi.org/10.1109/ICCSPP48568.2020.9182129>
- Wang, X. F., He, Z. H., Wang, K., Wang, Y. F., Zou, L., & Wu, Z. Z. (2023). A survey of text detection and recognition algorithms based on deep learning technology. *Neurocomputing*, 556, 126702. <https://doi.org/10.1016/j.neucom.2023.126702>
- Weng, Y., & Xia, C. (2019). A new deep learning-based handwritten character recognition system on mobile computing devices. *Mobile Networks and Applications*, 25(2), 402–411. <https://doi.org/10.1007/s11036-019-01243-5>
- Wu, C., Fan, W., He, Y., Sun, J., & Naoi, S. (2014). Handwritten character recognition by alternately trained relaxation convolutional neural network. *2014 14th international conference on frontiers in handwriting recognition* (pp. 291–296).