future internet

Article

A Fusion-Based Hybrid-Feature Approach for Recognition of Unconstrained Offline Handwritten Hindi Characters

Danveer Rajpal [1], Akhil Ranjan Garg [1], Om Prakash Mahela [2], Hassan Haes Alhelou [3] and Pierluigi Siano [4,5,*]

1 Department of Electrical Engineering, Faculty of Engineering, J.N.V. University, Jodhpur 342001, India; danveer.rajpal@rediffmail.com (D.R.); agarg@jnvu.edu.in (A.R.G.)
2 Power System Planning Division, Rajasthan Rajya Vidyut Prasaran Nigam Ltd., Jaipur 302005, India; opmahela@gmail.com
3 Department of Electrical Power Engineering, Tishreen University, Lattakia 2230, Syria; h.haesalhelou@gmail.com
4 Department of Management & Innovation Systems, University of Salerno, 84084 Fisciano, Italy
5 Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg 2193, South Africa
* Correspondence: psiano@unisa.it

Abstract: Hindi is the official language of India and used by a large population for several public services like postal, bank, judiciary, and public surveys. Efficient management of these services needs language-based automation. The proposed model addresses the problem of handwritten Hindi character recognition using a machine learning approach. The pre-trained DCNN models namely; InceptionV3-Net, VGG19-Net, and ResNet50 were used for the extraction of salient features from the characters' images. A novel approach of fusion is adopted in the proposed work; the DCNN-based features are fused with the handcrafted features received from Bi-orthogonal discrete wavelet transform. The feature size was reduced by the Principal Component Analysis method. The hybrid features were examined with popular classifiers namely; Multi-Layer Perceptron (MLP) and Support Vector Machine (SVM). The recognition cost was reduced by 84.37%. The model achieved significant scores of precision, recall, and F1-measure—98.78%, 98.67%, and 98.69%—with overall recognition accuracy of 98.73%.

Keywords: Bi-orthogonal; DCNN; DWT; Hindi characters; hybrid-features; fusion; MLP; PCA; SVM; transfer learning

Citation: Rajpal, D.; Garg, A.R.; Mahela, O.P.; Alhelou, H.H.; Siano, P. A Fusion-Based Hybrid-Feature Approach for Recognition of Unconstrained Offline Handwritten Hindi Characters. *Future Internet* **2021**, *13*, 239. https://doi.org/10.3390/fi13090239

Academic Editors: Jacopo Soldani and Stefano Forti

Received: 27 August 2021
Accepted: 14 September 2021
Published: 18 September 2021

1. Introduction

The increasing demand for the automation of language-based systems is high due to the associated vast application field. It includes digitalization and preservation of the manuscripts of historic significance, computerized editing of handwritten documents, automatic processing of cheques in the bank, recognition of postal address written on mails, parcels, etc. and their address-wise sorting through computer vision, translation of road safety-instructions written in the local language on roadside boards, computerized recognition of medical-aids as mentioned in handwritten prescription, and many more related applications. The machine-based recognition of handwritten scripts is much more difficult than that of printed ones due to inherent unconditional variation in shape, size, skewness, and degree of connectedness between various characters. Countries like India, China, Saudi Arabia, and the United Arab Emirates are developing automation systems in country-specific languages to serve its advantage to the mass of the people as large populations of these countries have not adopted English as their first language.

Many advancements have been reported for English language-based automation systems due to their global acceptance. Extra attention is needed for systems based on languages like Hindi (Devnagari), Chinese, Urdu, Farsi, etc., as they are in a developing phase. The proposed work is a small step in this direction for the Hindi script. The Hindi-consonant set is displayed in Figure 1.

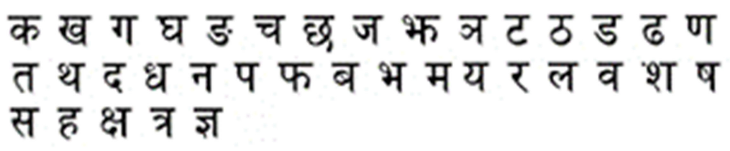
क ख ग घ ङ च छ ज झ ञ ट ठ ड ढ ण
त थ द ध न प फ ब भ म य र ल व श ष
स ह क्ष त्र ज्ञ

Figure 1. Consonant set of the Hindi script.

According to recent surveys [1–4] based on recognition of handwritten Hindi script, feature types and classification schemes can be broadly categorized in the following manner: Features are classified as structural and statistical. Structural features include lines, curves, loops, dots, end points, intersection points, edges, contours, and directional details. Sethi et al. [5] used features related to vertical–horizontal lines and left–right slants; Bhattacharya et al. [6] implemented stroke-based features; and Arora et al. [7] utilized horizontal-bar, vertical-bar, and intersection points as prime features.

Statistical features encompass zone-based, distance-based, pixel-density-based, histogram based, moment-based, and transformation-based features. Bajaj et al. [8] implemented density, moment, and descriptive-component-based features; Pal et al. [9] utilized chain code histogram as potential features; Pal et al. [10] received gradient-based directional features; Deshpande et. al. [11] developed coding scheme for connected segments of the characters; Iamsa et al. [12] examined the features collected from a histogram of gradients; Shitole et al. [13] estimated potential statistical features based on linear-discriminant-analysis (LDA) and PCA. Higher manual efforts are needed for extracting structural and statistical features; the usefulness of the features depends on the skill of the user.

The classification schemes can be broadly categorized as ANN-based, kernel-based, statistical-based, template-matching-based, and fuzzy-based. The ANN-based approach is well known for its potential of classifying non-linearly separable data; Rojatkar et al. [14] utilized MLP network with varying numbers of hidden units; Khanduja et al. [15] used feedforward neural network with backpropagation learning; Nikita Singh [16] implemented feedforward neural network with changing number of hidden layers. The kernel-based approach is useful in classifying higher dimensional data; Jangid et al. [17] employed an SVM classifier; Puri et al. [18] utilized a Gaussian-kernel-based support-vector-machine algorithm. Statistical-based approaches are popular due to the efficient mathematical modelling of the given classification problem; Sethi et al. [19] employed a decision-tree-search for classification; Parui et al. [20] implemented a hidden Markov model with Gaussian distribution; Holambe et al. [21] produced a comparative study of a KNN classifier by deciding the nearest neighbors on the basis of Euclidian-distance, cosine-similarity, etc. The template-matching approach was adopted by Pal et al. [22] in their comparative study of different classifiers. Hanmandlu et al. [23] implemented a fuzzy-logic-based classifier.

Advanced Deep Convolutional Neural Networks (DCNN) are trending for pattern recognition problems because of their excellent capabilities of auto-generative feature extraction and classification [24]. Deep networks need large-sized datasets for learning from scratch. Non-availability of such datasets restricts the use of DCNN models for several pattern recognition problems, including the proposed one. With transfer-learning approach, the trainable parameters of pre-trained deep networks are preserved and can be reutilized for a similar kind of pattern recognition problems, even with smaller-sized datasets [25]. However, the large-sized feature-vector of pre-trained DCNN models might result in increased classification costs.

1.1. Related Work

Several studies were relied upon for handcrafted features related to the structure of character image and/or associated statics. The main advantage of handcrafted features lies in their capability of customization. One can select a number of such features as per the problem of pattern recognition. For example, Verma [26] selected the features belonging to dots, lines, curves, and loops presented in the character image; Pal, Wakabayashi, and Kimura [22] focused on gradient and curvature-based features due to richness of the curves in the Hindi script; Dixit, Navghane, and Dandawate [27] took advantage of the time–frequency localization ability of DWT for salient feature extraction; Khanduja, Nain, and Panwar [15] developed a curve-fitting algorithm to receive curved-like features and combined it with the number of loops, intersection points, end points presented in the character image; Singh and Maring [28] worked on the features based on chain-code, zone-based-centroid, background-directional-distribution, and distance-profile; Jangid and Srivastava [17] addressed the problem of poor recognition of similar-shaped characters by focusing on critical regions of such characters and applied the Fisher discriminant function to extract features from these regions; Gupta, Sarkhel, Das, and Kundu [29] implemented opposition-based multi-objective Harmony Search algorithm to select features from local regions from character images. The handcrafted features need continuous human supervision and typical computations are associated with it.

CNN-based features are auto generative. The associated convolution and pooling operations make these features computationally-efficient. Sarkhel, Das, Das, Kundu, and Nasipuri [30] introduced multi-column multi-scale CNN architecture for feature extraction from character images; Chakraborty et al. [31] examined the effect of increasing convolutional layers on recognition accuracy. They also developed a hybrid model of CNN and Bi-directional Long Short-Term Memory (BLSTM) for the task; Jangid and Srivastava [32] adopted a layer-wise training scheme of CNN architecture; Sonawane and Shelke [33] implemented the transfer learning approach on a pre-trained DCNN model: Alexnet for character recognition. It can be observed from existing state-of-the-art methods related to CNN that the feature-vector sizes are large and might increase the classification cost.

The literature review reveals further research scope in the direction of exploring pre-trained DCNN models for feature extraction, classification-cost reduction, and experimentation with a combination of handcrafted and CNN-based features.

1.2. Motivation

The benefits of advanced deep-learning-based models and the associated limitations on the need of large-sized datasets and classification costs are the prime motivation for the proposed work. The main focus is on developing a low-cost recognition model with the advantage of excellent feature-extraction capabilities of advanced pre-trained DCNN models for the recognition of handwritten Hindi characters. The main motivations for the proposed work are as follows:

1. To introduce the novel approach of handwritten Hindi character recognition with the benefits of features received from pre-trained DCNN models, accompanied by the reduced computational loads of the classifiers.
2. The majority of previously reported works have been solely based on either handcrafted features or CNN-based features. No work has been reported yet in which both types of features are used in a single feature-vector for the recognition of handwritten Hindi characters. CNN-based features and handcrafted features have their own advantages—the former are auto-generative and the latter are rich in customization.
3. The model performance for each character-class of Hindi script has also not been covered in detail, such as character-wise correct and incorrect predictions; the amount of false-positive and false-negative predictions out of the total incorrect predictions and development of a confusion-matrix for all the 36 classes of Hindi consonants, etc.
4. The examination of the effectiveness of individual feature-types and their all-possible combinations is also a novel approach in relation to handwritten Hindi characters.

1.3. Contribution

The discussed limitations have been considered as an opportunity in the proposed work and efforts have been made to address them. The main contributions are as follows:

1. The scheme exploited pre-trained DCNN models, namely Inception-Net, VGG-NET, and Res-Net for feature extraction from the handwritten character-images; due to their excellent performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) annual competition [34].
2. The model experimented with the fresh approach of feature–fusion, where the features received from pre-trained DCNN models were fused with the features received from handcrafted methods. In the proposed scheme, Bi-orthogonal Discrete Wavelet Transform (BDWT) was the natural choice for handcrafted features because of its properties like flexibility, separability, scalability, and transformability with the power of multi-resolution analysis. A very limited work has been reported on the use of DWT for the recognition of handwritten Hindi characters [27,35]. The effective PCA method was implemented in the proposed work for dimensionality reduction of feature-vectors by preserving most of the important details. This helped in achieving a low computational cost.
3. The proposed scheme has thoroughly investigated the performance of the model for individual character-class. The number of performance metrics like precision, recall, and F1-measure was evaluated for the test samples of each character class to determine the amount of correct and incorrect recognitions. A confusion matrix was generated for precise character-wise result analysis.
4. The strength of individual feature-types present in the hybrid-feature-vector and their all-possible combinations were evaluated for recognition accuracy.
5. The proposed features were examined with two popular classifiers, namely MLP and SVM. This was done to examine the performance of proposed features over ANN-based and kernel-based approaches, respectively, for the given multiclass problem.
6. Various timings, related to feature-extraction and character-recognition, were estimated in the proposed work.

The rest of the paper is organized as follows: Section 2 contains the preliminary; Section 3 contains the methodology (method); Section 4 contains the results and discussions; Section 5 contains the conclusions.

2. Preliminary

This section will cover a relevant theoretical base of the techniques implemented in the proposed work.

2.1. Transfer Learning

Transfer learning is a technique of conveying knowledge received in one domain to another related domain. With respect to DCNN models, transfer learning is a method in which the weights learned by a model on a certain dataset can be reused for other similar kind of datasets. Such techniques are useful in receiving the advantages of deep learning-based models with limited data and less training time for pattern recognition problems [36]. The transfer learning approach has been successfully implemented in several recent studies based on medical applications [37–39]. In the proposed work, the transfer learning approach was implemented on pre-trained DCNN models, namely VGG19-Net, InceptionV3-Net, and ResNet-50 for feature extraction from the character images. The key properties and architecture of these models are discussed below.

2.1.1. VGG-19Net

The VGG19-Net has a total of 19 trainable layers, out of which 16 are convolutional layers and 3 are dense layers. The network has 5 Max-pooling layers, one after each convolutional block to handle feature redundancy. The input size is fixed as $224 \times 224 \times 3$. The size of the feature-vector is 4096 [40].

2.1.2. Inception V3-Net

The InceptionV3-Net has three types of inception modules that are basically well-designed convolution modules. These modules are used for generating distinct features with a reduced number of parameters. Several convolutional layers and pooling layers are arranged in parallel in each inception module. The input size is fixed as 299 × 299 × 3. The size of the feature vector is 2048 [41].

2.1.3. ResNet-50

ResNet was the winner of the ILSVRC competition in 2015 [42]. ResNet-50 is a 50-layer deep network with 49 convolutional layers and 1 dense layer. It is similar to the other networks having convolution, pooling, activation, and dense layers in cascade, but with the presence of an identity connection between the layers to skip one or more layers. The model assumes that residual learning is more effective than stacked learning. The input size is fixed as 224 × 224 × 3. The size of the feature vector is 2048.

Table 1 summarizes the architecture of pre-trained networks used in the proposed work.

Table 1. Summary of pre-trained network architecture.

Network	Depth	Input Size	Feature-Vector Size
VGG19-Net	19	224 × 224 × 3	4096
Inception V3-Net	48	299 × 299 × 3	2048
ResNet-50	50	224 × 224 × 3	2048

2.2. *Handcrafted Features: Bi-Orthogonal Discrete Wavelet Transform*

For Multi-Resolution Analysis (MRA) [43] of an image, Bi-orthogonal wavelets are preferable over orthogonal ones, due to their flexible nature of transformation—they can be invertible without being necessarily orthogonal. The Bi-orthogonal wavelets have in-built properties of symmetry, linear-phase, compact-support, and de-noising. The symmetry and linear phase ensure distortion-less wavelet coefficients; compact support is a guarantee of capturing minute details of a given pattern; the potential of de-noising can improve the recognition ability [44].

Bi-orthogonal wavelets can be used to develop symmetrical wavelet functions [45], which are useful in the effective edge-representation of images. Since edge representation is a significant aspect in the recognition of handwritten characters, the BDWT becomes the natural choice for handcrafted feature extraction in the proposed scheme.

The symmetry property needs exact reconstruction of the Bi-orthogonal filter-banks design [46]. The design carries analysis and synthesis filter banks. The scaling and transformation functions associated with the analysis filter bank are given by Equations (1) and (2):

$$\varnothing(x) = \sqrt{2}\sum_k L(k)\ \varnothing(2x - k) \tag{1}$$

$$\varphi(x) = \sqrt{2}\sum_k H(k)\ \varnothing(2x - k) \tag{2}$$

Here, L(k) and H(k) are responses of filters associated with scaling and transformation functions. For the synthesis-filter bank, the scaling and transformation functions are given by (3) and (4):

$$\widetilde{\varnothing}(x) = \sqrt{2}\sum_k \overline{L}(k)\ \widetilde{\varnothing}(2x - k) \tag{3}$$

$$\widetilde{\varphi}(x) = \sqrt{2}\sum_k \overline{H}(k)\widetilde{\varnothing}(2x - k) \tag{4}$$

Here, $\overline{L}(k)$ and $\overline{H}(k)$ are responses of filters associated with scaling and transformation functions of the synthesis-filter bank. The (3) and (4) are dual of (1) and (2) [47]. For exact reconstruction, the filter responses must satisfy the conditions given in (5), (6) and (7) [48]:

$$L(k)\ \overline{L}(k) = H(k)\ \overline{H}(k) = 1 \tag{5}$$

$$L(k)\ \ \overline{H}(k) = H(k)\ \overline{L}(k) = 0 \tag{6}$$

$$\overline{L}(k)\ L(k) + \ \overline{H}(k)\ H(k) = 1 \tag{7}$$

Referring to Equations (1)–(7), all the equations are linear in nature, ensuring a simple design and less complexity.

2.3. Principal Component Analysis

The PCA is an efficient method of minimizing data-redundancy; it can eliminate multiple collinearities present in the data. It offers flexibility in the selection of the desired number of reduced features in such a way that they carry most of the information of original datasets. The dimensionality reduction provides the benefits of fast processing, ease in visualizing and analyzing the data, and low computational cost of the model [49]. All these benefits make the PCA a genuine choice for the proposed work. The mathematics involved in finding principal components is as follows [13].

Suppose we have a dataset with N number of samples as given in (8):

$$D = [d_1, d_2, d_3 \ldots\ d_N] \tag{8}$$

where each sample carries F number of features as given by (9).

$$d_i = [{d_i}^1, {d_i}^2, {d_i}^3 \ldots\ {d_i}^F] \tag{9}$$

The covariance matrix for given dataset is determined by (10):

$$C = \frac{1}{N}\sum_{j=1}^{N}\left(d_j - \overline{d_j}\right)\cdot\left(d_j - \overline{d_j}\right)^T \tag{10}$$

Here $\overline{d_j}$ is the mean of data. With the help of covariance-matrix Eigen, values and Eigen-vectors are computed as given in (11):

$$CV = \lambda V \tag{11}$$

where C is a covariance matrix, V is Eigen vector and λ is Eigen value.

2.4. Multi Layer Perceptron

It is a class of Feed Forward Neural Networks (FFNN) with three types of layers, namely Input, Hidden, and Output. Depending on the applications, the network may have more than one hidden layer. The hidden units run on a nonlinear activation function and play an important role in solving the problems of classification and regression. The network uses backpropagation algorithms for its learning. The MLP network has a tremendous ability to solve pattern recognition problems [50]. The network architecture with one hidden layer is shown in Figure 2.

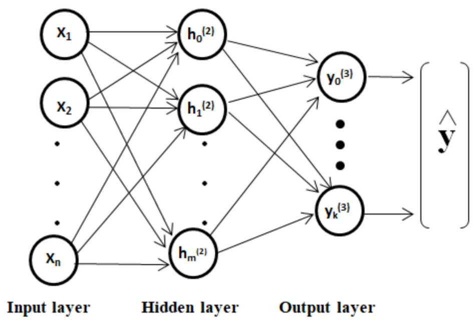

Figure 2. Architecture of the MLP network with a single hidden layer.

In the proposed work, Rectified Linear Unit (ReLU) activation function was used in hidden units of the MLP network, due to its low computational cost over Sigmoid and tanh functions. For hidden-unit *k*, the ReLU-activation function is given by (12):

$$h_k(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{12}$$

Here x is input to hidden-unit *k*.

In the proposed work, the Adam optimizer was used because of its ability to deal with sparse gradient and non-stationary objectives. These abilities are adopted from AdaGrad and RMSProp optimizers. The Adam solver is used to update the weight parameters in the following manner [51].

In the first step, gradient (g) is computed w.r.t. time instant t as:

$$g_t = \text{grad}(\theta_{t-1}) \tag{13}$$

where, θ represents weights and biases parameters. The g represents *dw* and *db*, respectively. In step 2, the first moment m_t and the second moment v_t are updated as:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{14}$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{15}$$

where, β_1 and β_2 are exponential decay rates ranging between 0 and 1. In step 3, bias-correction is applied on m_t and v_t as:

$$\hat{m}_t = \frac{m_t}{(1 - \beta_1^t)} \tag{16}$$

$$\hat{v}_t = \frac{v_t}{(1 - \beta_2^t)} \tag{17}$$

In the final step, the parameter update is given by:

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} \tag{18}$$

These steps are repeated till the final convergence.

2.5. *Support Vector Machine*

An SVM algorithm is used to find the best possible hyper-plane in K-dimensional feature space to separate the samples of different classes. The SVM classifier is popular because of its potential in solving for the global minimum. It can impressively control the over-fitting problem due to its feature generalization ability [52]. SVM can efficiently solve the classification problem of non-linearly separable data with kernel approach. All these benefits of SVM make it the prime choice for classification tasks in the proposed work. The mathematical modelling of the Kernel approach is given below:

The optimum hyperplane derived by the SVM classifier can be given by Equation (19):

$$\sum_{i=1}^{N} \alpha_i y_i x^T x_i + b = 0 \tag{19}$$

where, N represents the number of samples, α represents a Lagrange multiplier, and y_i represents the labelled value of i^{th} support vector, given as $y_i \in \{1, -1\}$ and b represents bias. To classify the nonlinearly-separable data using a linear decision-plane, the kernel approach can be used in the following manner [53].

With the help of Basis function φ, the given data is transformed to higher dimensions. By using Equation (19), the hyper plane in a higher dimension can be given as:

$$\sum_{i=1}^{N} \alpha_i y_i \ \varphi(x)^T \varphi(x_i) + b = 0 \tag{20}$$

Equation (20) can be rewritten as:

$$\sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + b = 0 \tag{21}$$

Here, K (x, x_i) is kernel function, which is defined as:

$$K(x_i, x_j) = \varphi(x_i)\varphi(x_j) \tag{22}$$

In the proposed work, the Linear kernel was used because of its ability to make SVM learn fast, the least risk of over-fitting, and the need for zero hyper parameters [54]. It is given by:

$$K\ (x_i, x_j) = x_i \cdot x_j \tag{23}$$

3. Research Method

The design of the proposed scheme is displayed in Figure 3. The character image was supplied for individual feature-extraction schemes. In the DWT-based scheme, the BDWT was applied on a given image to collect a feature-vector in the form of wavelet coefficients. The same image was given to individual pre-trained DCNN models (VGG-19Net, ResNet-50, and InceptionV3-Net) to receive respective feature-vectors. At the end of the feature-extraction stage, we received four different feature-vectors related to the given image; they are labelled as F1, F2, F3, and F4. Next is the dimensionality reduction stage, where the PCA method was used to reduce the size of individual feature-vectors; they were reduced to equal sizes and marked as W, V, R, and I. The reduced-sized feature-vectors were supplied to the feature-fusion stage, where hybrid feature-vectors were developed with all possible combinations of W, V, R, and I. At this stage, 15 new datasets (D1 to D15) were created by collecting respective hybrid feature-vectors for all the images of input dataset. The performance of individual datasets was estimated over MLP and SVM classifiers. The respective results were collected and analyzed in the output stage.

The experimental framework was developed in a Python environment. Several open-source libraries were used for the task: open-cv, python-imaging-library, py-wavelet, keras, numpy, pandas, scikit-learn, seaborn, python-time-module, etc. The experiments were simulated on the powerful Google Co-laboratory platform. Co-laboratory is supported with tesla k-80, 2496 CUDA cores, 12 GB GDDR5-VRAM GPU, hyper-threaded Xeon processor, 12.6 GB RAM, and 33 GB storage capacity.

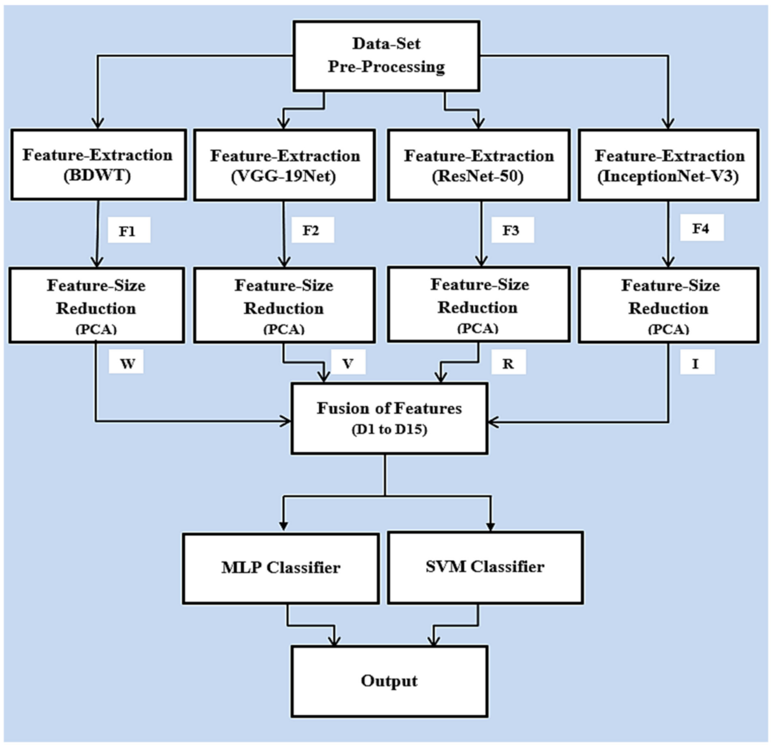
Data-Set Pre-Processing
Feature-Extraction (BDWT)
Feature-Extraction (VGG-19Net)
Feature-Extraction (ResNet-50)
Feature-Extraction (InceptionNet-V3)
F1
F2
F3
F4
Feature-Size Reduction (PCA)
Feature-Size Reduction (PCA)
Feature-Size Reduction (PCA)
Feature-Size Reduction (PCA)
W
V
R
I
Fusion of Features (D1 to D15)
MLP Classifier
SVM Classifier
Output

Figure 3. Design of the proposed work.

3.1. *Dataset Pre-Processing*

A dataset of Devnagari characters was prepared by collecting handwritten documents from a wide variety of individuals belonging to different age groups and professions. Individual characters were scanned and cropped manually. The whole dataset was prepared and made available in the public domain by Acharya, Pant, and Gyawali [55]. A dataset of size 18,000 was prepared for the proposed work; it included all the 36 consonants. All the images were checked for a uniform size of 32 by 32 and converted into a grey-level format.

3.2. *Feature Extraction*

In the proposed scheme, multiple approaches were applied for the extraction of features from the character images. The approaches are described in the following sub-sections.

3.2.1. Discrete Wavelet Transform

The Bi-orthogonal Discrete Wavelet Transform (BDWT) was applied to individual images for receiving salient features in terms of wavelet coefficients. The two-dimensional coefficients, including approximation, horizontal, vertical, and diagonal details, were transformed into a feature-vector, as shown in Figure 4. This feature-vector was marked as F1. In the proposed work, Bi-orthogonal-1.3 wavelet (Bior-1.3) was used up to second-level decomposition of character image. The kernel size of the Bior-1.3 wavelet is summarized in Table 2.

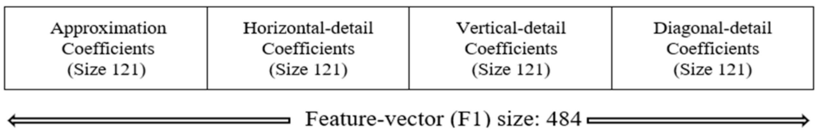
Approximation Coefficients (Size 121)
Horizontal-detail Coefficients (Size 121)
Vertical-detail Coefficients (Size 121)
Diagonal-detail Coefficients (Size 121)
Feature-vector (F1) size: 484

Figure 4. The frame format of features received from DWT.

Table 2. Kernel size of the Bior-1.3 wavelet.

Sl. No.	Transformation Level	Kernel Size
1	I	(18 × 18)
2	II	(11 × 11)

Referring to Table 2, the level 1 decomposition produced the feature vector size of 1296 (i.e., 4×(18 × 18)), including all the four types of coefficients (approximation, horizontal, vertical, and diagonal), while the feature-vector size of 484 (i.e., 4×(11 × 11)) for level-2 decomposition offered a promising reduction in feature dimensionality. This made the level 2 decomposition a logical choice for the proposed work.

3.2.2. Pre-Trained Deep Convolutional Network

In the proposed scheme, three pre-trained DCNN models, namely VGG-19Net, ResNet-50, and InceptionV3-Net were individually used for feature extraction from character images. The trainable layers of individual models were frozen. The input images were resized as per the need of the models (refer to Table 1). The feature-vector for each image was collected at the final global average pooling layer of the respective models. The corresponding feature-vectors are shown in Figure 5 and marked as F2, F3, and F4.

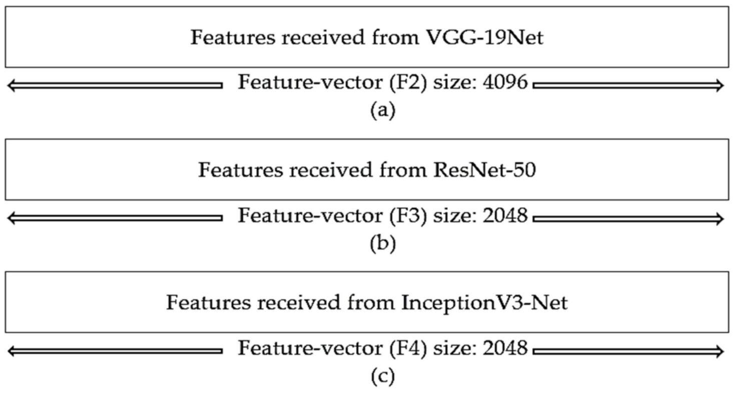
Features received from VGG-19Net
Feature-vector (F2) size: 4096
(a)
Features received from ResNet-50
Feature-vector (F3) size: 2048
(b)
Features received from InceptionV3-Net
Feature-vector (F4) size: 2048
(c)

Figure 5. The feature-vector formats received from (**a**) VGG19Net; (**b**) Resnet-50; (**c**) Inception V3-Net.

3.3. Feature-Vector Size Reduction

The Principal Component Analysis method was used for reducing the size of various feature vectors. The trial-and-error method was adopted to decide the number of PCA components. Initially, 20 PCA components were estimated from individual feature-vectors (i.e., from F1, F2, F3 and F4). The respective principal components were fused into a single feature-vector. A sub-dataset of 3600 such fused feature-vectors (i.e., 100 samples from each character-class) was prepared for the purpose. The sub-dataset was used to train and test the proposed classifiers. The process was repeated with the sub-dataset by increasing the number of PCA components up to 60 in the step of 10. It was observed that recognition accuracy improved in a good amount by increasing PCA components in the range of 20 to 40, but no significant improvements were noticed for the range of 40 to 60. This made the 40 PCA components an optimum choice. In the proposed work, the size of individual feature-vectors (F1, F2, F3, and F4), were reduced to 40 and the resultant vectors were marked as W, V, R, and I, respectively.

3.4. Fusion of Features

Several hybrid feature-vectors were developed with all the possible combinations of reduced feature-vectors W, V, R, and I. The 11 combinations were based on fusion of two or more feature-types, while 4 combinations have no fusion (refer to Table 3). The format of all hybrid feature-vectors is given in Figure 6. The respective hybrid feature-vectors were derived for all the character images of the input dataset. At this stage, 15 new datasets were produced; they are summarized in Table 3.

Table 3. Summary of various datasets used in the proposed work.

Dataset No.	Hybrid Feature-Vector Category	Source	Feature-Vector Size
D1	Individual type (no fusion)	BDWT (W)	40
D2		VGG-19Net (V)	40
D3		ResNet-50 (R)	40
D4		InceptionNet-V3 (I)	40
D5	Hybrid type (fusion of two types of features)	Fusion of W and V	80
D6		Fusion of W and R	80
D7		Fusion of W and I	80
D8		Fusion of V and R	80
D9		Fusion of V and I	80
D10		Fusion of R and I	80
D11	Hybrid type (fusion of three types of features)	Fusion of W, V and R	120
D12		Fusion of W, R and I	120
D13		Fusion of W, V and I	120
D14		Fusion of V, R and I	120
D15	Hybrid type (fusion of four types of features)	Fusion of W, V, R and I	160

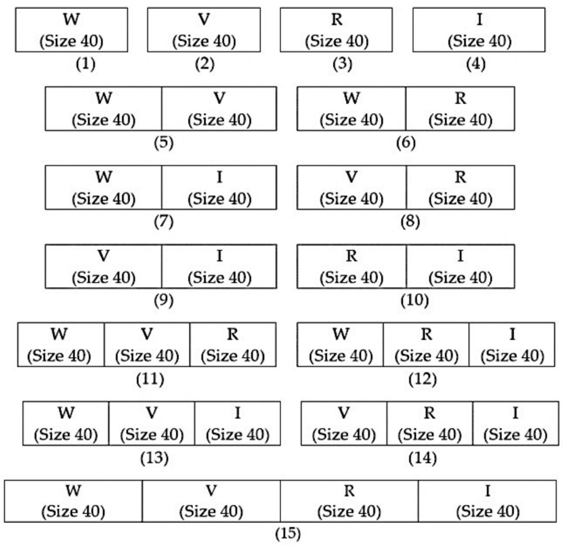
W (Size 40) (1)
V (Size 40) (2)
R (Size 40) (3)
I (Size 40) (4)
W (Size 40) V (Size 40) (5)
W (Size 40) R (Size 40) (6)
W (Size 40) I (Size 40) (7)
V (Size 40) R (Size 40) (8)
V (Size 40) I (Size 40) (9)
R (Size 40) I (Size 40) (10)
W (Size 40) V (Size 40) R (Size 40) (11)
W (Size 40) R (Size 40) I (Size 40) (12)
W (Size 40) V (Size 40) I (Size 40) (13)
V (Size 40) R (Size 40) I (Size 40) (14)
W (Size 40) V (Size 40) R (Size 40) I (Size 40) (15)

Figure 6. The format of hybrid feature-vectors; (1) to (4) are feature-vectors without fusion and they are related to the new datasets—D1 to D4, respectively; (5) to (15) are hybrid feature-vectors with fusion of 2 or more feature-types and they belong to new datasets, D5 to D15, respectively.

3.5. *Classification*

The newly created datasets (D1 to D15) were examined with MLP and SVM classifiers. Both the classifiers were trained and tested over these datasets individually. The k cross-validation scheme was adopted for the classification task to get generalized results and to avoid over-fitting problems that might be caused a moderate-sized dataset. In this scheme, each sample of the dataset was used one times as a test sample and k-1 times as a training sample. The general rule and empirical experience suggested the most preferable values of k to be 5 or 10 [56]. We selected k as 5 i.e., a 5-cross validation scheme in the present work. In the Result and Discussion section, the mean value of the results received from the five cross-validation schemes has been mentioned by default. The major specifications of both the classifiers used in the proposed scheme are summarized in Tables 4 and 5.

Table 4. Specifications of an MLP classifier as used in the proposed work.

Sl. No.	Parameter	Specification
1	Size of input layer	40 (Datasets 1–4), 80 (Datasets 5–10), 120 (Datasets 11–14) and 160 (Dataset 15)
2	No. of hidden layers	1
3	Units in hidden layer	36 (Datasets 1–4), 58 (Datasets 5–10), 74 (Datasets 11–14) and 85 (Datasets 15)
4	Activation function in hidden layer unit	Rectified linear unit
5	Size of output layer	36
6	Solver	Adaptive moment
7	Learning rate	0.001
8	Number of iterations	500

Table 5. Specifications of an SVM classifier as used in the proposed work.

Sl. No.	Parameter	Specification
1	Regularization parameter	1.0
2	Kernel-cache size	200 MB
3	Shape of decision function	One Versus Rest
4	Type of kernel	Linear
5	Stopping-criterion tolerance	0.001
6	No. of iteration	−1 (no limit)
7	Break ties	False
8	Probability	True
9	Shrinking	True

The number of hidden units was selected by experimenting with the values in the range of 30 to 100 for different input sizes, as mentioned in Table 4. Concerning optimal results obtained, the hidden units were selected as 36, 58, 74, and 85 for respective input sizes.

3.6. Performance Metrics

The results were compiled in terms of recognition accuracy, precision, recall, and F1-measure. These terms were determined with the help of True Positive (TP), True Negative (TN), False Negative (FN), and False positive (FP) predictions, made by the classifiers. The effectiveness of the proposed feature-extraction techniques was visualized with the help of the Kernel Density Estimation (KDE) algorithm.

The mathematical formulations used for the estimation of the mentioned metrics are given in Equations (24)–(27):

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{24}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{25}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{26}$$

$$\text{F1} - \text{score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{27}$$

4. Results and Discussions

The various results of the proposed scheme are compiled in this section with a detailed discussion on critical observations.

4.1. Results

The results of the proposed scheme are summarized in Table 6 in terms of overall recognition accuracy achieved by the model for all the 15 types of datasets. It can be observed from Table 6 that both the classifiers produced the highest recognition accuracy for Dataset 15. Dataset 15 was developed by fusion of features received from Wavelets, VGG-19Net, InceptionV3-Net, and ResNet-50.

Table 6. Recognition accuracy achieved by the model from proposed feature-extraction schemes.

Sl. No.	**Dataset**	**Recognition Accuracy (%)**	
		MLP Classifier	**SVM Classifier**
1	D1	84.70	75.37
2	D2	**86.85**	82.44
3	D3	86.59	**82.66**
4	D4	82.22	79.44
5	D5	94.07	92.14
6	D6	94.33	91.85
7	D7	92.33	90.55
8	D8	**95.37**	**93.92**
9	D9	95.22	92.11
10	D10	94.59	92.44
11	D11	97.55	96.77
12	D12	97.03	96.66
13	D13	97.25	96.37
14	D14	**98.03**	**97.51**
15	D15	**98.73**	**98.18**

The best results related to the individual hybrid-feature-vector categories (refer to Table 3) are bold-faces in Table 6 for the purpose of critical discussions. Figure 7 shows a comparative analysis of the results, achieved by the two classifiers from 15 types of datasets, as mentioned in Table 6.

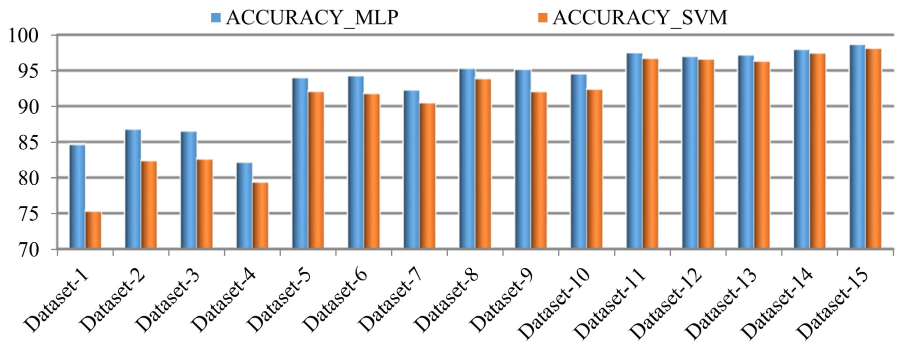

Figure 7. Comparative analysis of recognition accuracy achieved by MLP and SVM classifiers for proposed feature-extraction techniques.

The individual character-wise results in terms of precision, recall, and F1 score are compiled in Tables 7 and 9–12. For readability purposes, the values were rounded off to two decimal places. For each character class in Tables 9 and 12, the cell with the highest score was shaded for further analysis.

It has been observed from Tables 7 and 9–12 that the proposed scheme achieved optimum results from Dataset 15 (i.e., D15). The overall recognition accuracy achieved with Dataset 15 was 98.73% and 98.18% in relation to the MLP and SVM classifiers, respectively. The mean values of precision, recall, and F1 score for Dataset 15 were 98.78%, 98.67%, and 98.69% on the MLP classifier; and 98.11%, 98.22%, and 98.25% on the SVM classifier. The Dataset 15 has been considered for various comparative analysis, generation of confusion matrix, and visualization of feature-separation in the upcoming sections. Figure 8 shows a character-wise comparative analysis of the two classifiers for Dataset 15 in terms of precision, recall, and F1 score.

Table 7. Character-wise precision produced by an MLP classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
0	क	0.93	0.97	0.90	0.83	0.97	0.97	0.96	0.99	0.96	1.00	1.00	0.97	1.00	0.99	1.00	0.96
1	ख	0.78	0.91	0.86	0.92	0.93	0.99	0.93	0.94	0.96	0.99	0.95	1.00	0.95	0.98	0.99	0.94
2	ग	0.88	0.87	0.85	0.96	0.92	0.99	0.88	0.97	0.97	0.97	0.97	1.00	0.99	1.00	0.99	0.95
3	घ	0.71	0.92	0.80	0.68	0.96	0.90	0.87	0.94	0.92	0.89	0.97	0.88	0.93	0.99	0.97	0.89
4	ङ	0.85	0.80	0.93	0.68	0.94	0.94	0.85	0.91	0.91	0.93	0.97	0.95	1.00	0.99	0.99	0.91
5	च	0.91	0.99	0.96	0.79	0.96	1.00	0.99	1.00	0.95	0.97	1.00	1.00	1.00	0.99	1.00	0.97
6	छ	0.90	0.83	0.85	0.90	0.99	0.94	0.95	0.96	0.98	0.91	0.98	0.98	0.96	1.00	0.99	0.94
7	ज	0.89	0.93	0.89	0.82	0.95	0.97	0.97	0.97	0.96	0.99	0.99	0.99	0.97	1.00	0.99	0.95
8	झ	0.96	0.95	0.96	0.89	0.95	0.99	0.95	0.97	1.00	0.97	1.00	1.00	0.98	1.00	1.00	0.97
9	ञ	0.91	0.91	0.94	0.89	0.96	0.94	0.96	0.98	1.00	0.98	0.96	0.99	0.99	1.00	0.99	0.96
10	ट	0.87	0.91	0.86	0.90	1.00	0.95	0.94	0.97	0.99	0.94	1.00	0.98	0.99	1.00	0.99	0.95
11	ठ	0.90	0.94	0.90	0.86	0.91	0.96	0.94	0.98	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.96
12	ड	0.87	0.84	0.87	0.78	0.93	0.89	0.91	0.96	0.99	0.94	0.97	0.94	0.97	0.99	0.99	0.92
13	ढ	0.85	0.78	0.89	0.90	0.94	0.93	0.95	0.94	0.94	0.94	0.97	0.97	0.99	0.93	0.99	0.93
14	ण	0.86	0.95	0.96	0.88	0.92	0.98	0.97	0.99	1.00	1.00	0.97	0.98	0.99	1.00	0.99	0.96
15	त	0.95	0.95	0.96	0.77	0.97	0.94	0.96	0.95	0.93	0.96	0.97	1.00	0.99	0.97	0.99	0.95
16	थ	0.90	0.84	0.84	0.74	0.97	0.93	0.94	0.95	0.94	0.90	1.00	0.96	0.97	0.98	0.98	0.92
17	द	0.91	0.85	0.82	0.75	0.85	0.95	0.89	0.96	0.87	0.85	0.92	0.94	0.94	0.95	0.98	0.90
18	ध	0.80	0.84	0.84	0.85	0.89	0.93	0.92	0.93	0.95	0.91	0.96	0.97	0.95	0.93	0.98	0.91
19	न	0.91	0.84	0.90	0.79	0.95	0.95	0.88	0.95	0.91	0.92	0.97	1.00	0.99	0.95	1.00	0.93
20	प	0.85	0.91	0.86	0.82	0.93	0.93	0.91	0.98	0.96	0.96	1.00	1.00	0.98	1.00	1.00	0.94
21	फ	0.90	0.94	0.92	0.81	0.99	0.99	0.95	0.98	0.97	0.99	1.00	0.99	0.98	0.96	1.00	0.96
22	ब	0.74	0.85	0.86	0.81	0.90	0.92	0.93	0.91	0.89	0.92	0.96	0.99	0.95	1.00	0.98	0.91
23	भ	0.82	0.78	0.79	0.76	0.89	0.93	0.92	0.90	0.90	0.91	1.00	0.97	0.97	0.99	0.98	0.90
24	म	0.81	0.83	0.81	0.86	0.91	0.96	0.91	0.93	0.95	0.98	0.96	0.94	0.94	0.97	0.99	0.92
25	य	0.80	0.87	0.83	0.83	0.89	0.92	0.93	0.90	0.92	0.88	0.92	0.94	0.93	0.96	0.96	0.90
26	र	0.90	0.88	0.89	0.72	0.96	0.93	0.95	0.99	0.99	0.98	0.99	0.99	1.00	1.00	0.99	0.94
27	ल	0.89	0.89	0.87	0.85	0.99	0.93	0.91	0.99	0.98	0.97	0.97	1.00	0.98	1.00	0.99	0.95
28	व	0.66	0.74	0.85	0.79	0.87	0.86	0.85	0.91	0.93	0.96	0.97	0.96	0.93	0.99	0.97	0.88
29	श	0.81	0.87	0.85	0.84	1.00	0.92	0.95	0.97	0.95	0.99	1.00	0.97	0.99	1.00	0.99	0.94
30	ष	0.81	0.85	0.87	0.84	0.99	0.96	0.90	0.97	1.00	0.93	0.99	0.98	0.99	0.99	0.99	0.94
31	स	0.80	0.76	0.76	0.81	0.95	0.90	0.88	0.92	0.93	0.94	1.00	0.92	0.96	0.99	0.98	0.90
32	ह	0.82	0.88	0.73	0.94	0.87	0.92	0.85	0.96	0.89	0.91	0.94	0.96	0.95	0.89	0.98	0.90
33	क्ष	0.77	0.82	0.88	0.77	0.97	0.95	0.89	0.92	0.99	0.92	0.99	0.89	0.97	0.95	0.99	0.91
34	त्र	0.75	0.82	0.86	0.88	0.95	0.97	0.92	0.95	0.98	0.97	1.00	0.99	0.99	0.99	0.99	0.93
35	ज्ञ	0.96	0.79	0.86	0.76	0.96	0.97	0.95	0.95	0.95	0.95	0.95	0.97	0.99	1.00	0.98	0.93

Table 8. Character-wise recall produced by an MLP classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
0	क	0.89	0.90	0.93	0.74	0.97	0.96	0.95	0.99	0.99	0.97	1.00	1.00	0.98	0.96	1.00	0.95
1	ख	0.86	0.90	0.92	0.90	0.92	0.95	0.95	0.96	1.00	0.99	1.00	0.97	0.99	0.96	0.99	0.95
2	ग	0.83	0.87	0.91	0.84	0.97	0.94	0.98	0.98	0.99	0.98	0.99	0.94	0.99	1.00	0.99	0.95
3	घ	0.81	0.90	0.79	0.77	0.91	0.92	0.87	0.93	0.99	0.88	0.98	0.91	0.93	0.96	0.97	0.90
4	ङ	0.86	0.93	0.90	0.85	0.97	0.89	0.90	0.99	0.95	0.97	0.94	0.98	0.96	0.97	0.98	0.94
5	च	0.86	0.89	0.91	0.89	0.94	0.97	0.97	0.97	0.98	0.97	0.99	0.99	1.00	0.99	1.00	0.95
6	छ	0.87	0.83	0.91	0.85	0.95	0.94	0.95	0.97	0.95	0.84	0.95	0.95	0.96	0.95	0.98	0.92
7	ज	0.90	0.83	0.82	0.89	0.93	0.97	1.00	1.00	0.97	0.93	0.96	0.99	0.97	0.99	0.99	0.94
8	झ	0.97	0.96	0.96	0.88	0.98	0.99	0.97	0.95	0.97	0.99	1.00	0.99	1.00	1.00	1.00	0.97

Table 8. *Cont.*

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
9	ञ	0.89	0.93	0.83	0.85	0.94	0.97	0.91	0.95	0.94	0.97	0.99	0.96	0.99	0.99	0.99	0.94
10	ट	0.92	0.91	0.98	0.80	0.94	0.94	1.00	0.99	0.95	0.99	0.98	0.98	0.99	1.00	0.99	0.96
11	ठ	0.83	0.89	1.00	0.98	0.97	0.96	0.97	0.97	0.99	0.94	1.00	1.00	0.99	1.00	1.00	0.97
12	ड	0.86	0.93	0.80	0.89	0.88	0.97	0.88	0.93	0.97	0.93	1.00	1.00	0.98	0.98	0.98	0.93
13	ढ	0.90	0.87	0.86	0.80	0.95	0.93	0.92	0.97	0.95	0.94	0.96	1.00	0.97	0.96	0.99	0.93
14	ण	0.83	0.90	0.92	0.95	0.96	0.95	0.97	0.98	0.97	0.97	0.99	1.00	0.97	0.97	1.00	0.96
15	त	0.94	0.88	0.90	0.83	0.99	0.97	0.93	0.99	0.96	0.99	0.98	1.00	0.97	0.99	1.00	0.95
16	थ	0.85	0.86	0.82	0.75	0.89	0.96	0.94	0.95	0.93	0.88	0.98	0.99	0.93	0.98	0.98	0.91
17	द	0.85	0.74	0.85	0.69	0.95	0.95	0.87	0.97	0.96	0.92	0.96	0.93	0.97	0.93	0.98	0.90
18	ध	0.77	0.90	0.86	0.83	0.95	0.94	0.91	0.94	0.95	0.93	0.96	0.93	0.96	0.97	0.98	0.92
19	न	0.83	0.85	0.88	0.78	0.97	0.95	0.91	0.96	0.92	0.99	0.99	0.97	0.99	0.96	0.99	0.93
20	प	0.78	0.92	0.92	0.81	0.94	0.96	0.91	0.97	0.93	0.96	0.96	0.99	0.98	1.00	0.99	0.93
21	फ	0.87	0.96	0.86	0.83	0.99	0.97	0.97	0.98	0.99	0.96	0.97	1.00	0.99	1.00	1.00	0.96
22	ब	0.74	0.83	0.88	0.81	0.91	0.95	0.91	0.86	0.96	0.95	0.94	0.95	0.97	1.00	0.97	0.91
23	भ	0.76	0.82	0.76	0.81	0.94	0.88	0.95	0.91	0.95	0.97	0.95	0.92	0.93	0.97	0.98	0.90
24	म	0.84	0.82	0.90	0.88	0.94	0.93	0.89	0.88	0.91	0.96	0.95	1.00	0.98	1.00	0.98	0.92
25	य	0.78	0.83	0.83	0.77	0.87	0.89	0.86	0.94	0.92	0.92	0.99	0.99	0.98	1.00	0.97	0.90
26	र	0.85	0.84	0.81	0.83	0.92	0.94	0.95	1.00	0.96	1.00	0.99	0.97	0.94	0.99	0.99	0.93
27	ल	0.94	0.91	0.94	0.85	0.98	1.00	0.95	0.95	0.92	0.97	0.97	0.99	1.00	1.00	0.99	0.96
28	व	0.78	0.85	0.82	0.72	0.81	0.89	0.88	0.97	0.87	0.93	0.97	0.96	0.93	0.98	0.98	0.89
29	श	0.93	0.87	0.89	0.83	0.94	0.95	0.96	0.90	0.91	0.87	0.97	0.95	1.00	0.96	0.99	0.93
30	ष	0.85	0.86	0.87	0.82	0.97	0.95	0.93	0.95	0.95	0.97	0.97	0.99	0.97	0.99	0.99	0.94
31	स	0.75	0.75	0.76	0.83	0.92	0.91	0.84	0.93	0.91	0.88	0.99	0.95	0.93	0.97	0.98	0.89
32	ह	0.82	0.77	0.75	0.73	0.96	0.89	0.78	0.92	0.92	0.85	0.94	0.93	0.97	1.00	0.99	0.88
33	क्ष	0.87	0.86	0.87	0.80	0.93	0.94	0.88	0.93	0.96	0.96	1.00	0.97	0.97	1.00	0.98	0.93
34	त्र	0.83	0.82	0.85	0.79	0.94	0.95	0.94	0.95	0.98	0.97	0.97	0.95	1.00	0.99	0.98	0.93
35	ज्ञ	0.83	0.86	0.87	0.75	0.95	0.94	0.91	0.95	0.96	0.97	0.99	0.96	1.00	0.97	0.98	0.93

Table 9. Character-wise F1 score produced by an MLP classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
0	क	0.91	0.94	0.92	0.78	0.97	0.96	0.95	0.99	0.98	0.98	1.00	0.98	0.99	0.98	1.00	0.96
1	ख	0.82	0.91	0.88	0.91	0.93	0.97	0.94	0.95	0.98	0.99	0.97	0.99	0.97	0.97	0.99	0.94
2	ग	0.85	0.87	0.88	0.90	0.94	0.96	0.93	0.98	0.98	0.98	0.98	0.97	0.99	1.00	0.99	0.95
3	घ	0.75	0.91	0.79	0.72	0.94	0.91	0.87	0.93	0.95	0.88	0.97	0.89	0.93	0.97	0.97	0.89
4	ङ	0.86	0.86	0.92	0.76	0.96	0.91	0.88	0.94	0.93	0.95	0.96	0.97	0.98	0.98	0.99	0.92
5	च	0.88	0.94	0.93	0.84	0.95	0.99	0.98	0.99	0.97	0.97	0.99	0.99	1.00	0.99	1.00	0.96
6	छ	0.89	0.83	0.88	0.87	0.97	0.94	0.95	0.97	0.97	0.87	0.97	0.97	0.96	0.98	0.98	0.93
7	ज	0.90	0.88	0.85	0.85	0.94	0.97	0.99	0.98	0.97	0.96	0.97	0.99	0.97	0.99	0.99	0.95
8	झ	0.97	0.95	0.96	0.88	0.96	0.99	0.96	0.96	0.99	0.98	1.00	0.99	0.99	1.00	1.00	0.97
9	ञ	0.90	0.92	0.88	0.87	0.95	0.96	0.93	0.96	0.97	0.97	0.97	0.97	0.99	0.99	0.99	0.95
10	ट	0.89	0.91	0.91	0.85	0.97	0.94	0.97	0.98	0.97	0.96	0.99	0.98	0.99	1.00	0.99	0.95
11	ठ	0.86	0.91	0.95	0.92	0.94	0.96	0.95	0.98	0.99	0.97	1.00	1.00	0.99	1.00	1.00	0.96
12	ड	0.86	0.88	0.83	0.83	0.90	0.93	0.90	0.95	0.98	0.94	0.98	0.97	0.97	0.98	0.98	0.93
13	ढ	0.87	0.82	0.88	0.85	0.95	0.93	0.93	0.95	0.94	0.94	0.97	0.98	0.98	0.95	0.99	0.93
14	ण	0.84	0.92	0.94	0.91	0.94	0.96	0.97	0.98	0.98	0.98	0.98	0.99	0.98	0.99	1.00	0.96
15	त	0.94	0.91	0.93	0.80	0.98	0.96	0.95	0.97	0.94	0.97	0.97	1.00	0.98	0.98	0.99	0.95
16	थ	0.88	0.85	0.83	0.74	0.93	0.95	0.94	0.95	0.93	0.89	0.99	0.97	0.95	0.98	0.98	0.92
17	द	0.88	0.79	0.84	0.72	0.90	0.95	0.88	0.97	0.92	0.88	0.94	0.93	0.96	0.94	0.98	0.90
18	ध	0.79	0.87	0.85	0.84	0.92	0.93	0.92	0.94	0.95	0.92	0.96	0.95	0.96	0.95	0.98	0.92
19	न	0.87	0.85	0.89	0.79	0.96	0.95	0.90	0.96	0.91	0.95	0.98	0.99	0.99	0.96	0.99	0.93
20	प	0.81	0.91	0.89	0.81	0.94	0.94	0.91	0.98	0.95	0.96	0.98	0.99	0.98	1.00	0.99	0.94
21	फ	0.89	0.95	0.89	0.82	0.99	0.98	0.96	0.98	0.98	0.98	0.98	0.99	0.98	0.98	1.00	0.96

Table 9. *Cont.*

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
22	ब	0.74	0.84	0.87	0.81	0.91	0.93	0.92	0.89	0.93	0.93	0.95	0.97	0.96	1.00	0.98	0.91
23	भ	0.79	0.80	0.77	0.78	0.91	0.91	0.93	0.90	0.92	0.94	0.98	0.95	0.95	0.98	0.98	0.90
24	म	0.83	0.82	0.85	0.87	0.93	0.94	0.90	0.90	0.93	0.97	0.96	0.97	0.96	0.99	0.98	0.92
25	य	0.79	0.85	0.83	0.80	0.88	0.91	0.90	0.92	0.92	0.90	0.95	0.96	0.95	0.98	0.96	0.90
26	र	0.87	0.86	0.85	0.77	0.94	0.94	0.95	0.99	0.97	0.99	0.99	0.98	0.97	0.99	0.99	0.94
27	ल	0.91	0.90	0.91	0.85	0.98	0.96	0.93	0.97	0.95	0.97	0.97	0.99	0.99	1.00	0.99	0.95
28	व	0.72	0.79	0.83	0.75	0.84	0.88	0.86	0.93	0.90	0.94	0.97	0.96	0.93	0.98	0.98	0.88
29	श	0.87	0.87	0.87	0.83	0.97	0.94	0.96	0.93	0.93	0.93	0.99	0.96	0.99	0.98	0.99	0.93
30	ष	0.83	0.86	0.87	0.83	0.98	0.95	0.92	0.96	0.98	0.95	0.98	0.98	0.98	0.99	0.99	0.94
31	स	0.77	0.75	0.76	0.82	0.93	0.91	0.86	0.93	0.92	0.91	0.99	0.93	0.94	0.98	0.98	0.89
32	ह	0.82	0.82	0.74	0.82	0.91	0.91	0.81	0.94	0.90	0.88	0.94	0.95	0.96	0.94	0.99	0.89
33	क्ष	0.82	0.84	0.88	0.79	0.95	0.94	0.88	0.93	0.97	0.94	0.99	0.93	0.97	0.97	0.98	0.92
34	त्र	0.79	0.82	0.85	0.83	0.95	0.96	0.93	0.95	0.98	0.97	0.99	0.97	0.99	0.99	0.99	0.93
35	ज्ञ	0.89	0.83	0.86	0.76	0.96	0.95	0.93	0.95	0.95	0.96	0.97	0.97	0.99	0.99	0.98	0.93

Table 10. Character-wise precision produced by an SVM classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
0	क	0.84	0.87	0.89	0.76	0.97	0.91	0.96	0.98	0.91	0.98	0.99	0.97	1.00	1.00	1.00	0.94
1	ख	0.69	0.79	0.83	0.80	0.91	0.90	0.92	0.94	0.91	0.96	0.99	0.97	0.95	0.99	0.99	0.90
2	ग	0.77	0.85	0.79	0.88	0.85	0.94	0.94	0.98	0.97	0.98	0.97	0.99	0.98	0.94	0.98	0.92
3	घ	0.54	0.73	0.65	0.52	0.84	0.80	0.77	0.83	0.79	0.78	0.92	0.86	0.89	0.96	0.93	0.79
4	ङ	0.70	0.76	0.83	0.66	0.86	0.90	0.84	0.90	0.90	0.88	0.99	0.95	0.98	0.95	0.98	0.87
5	च	0.86	0.94	0.88	0.83	0.96	0.97	0.97	0.96	0.95	0.90	0.98	0.99	1.00	0.99	1.00	0.95
6	छ	0.84	0.75	0.79	0.82	0.93	0.94	0.91	0.92	0.93	0.84	0.93	0.95	0.93	0.99	0.96	0.90
7	ज	0.83	0.83	0.85	0.81	0.95	0.99	0.95	0.93	0.90	0.95	0.97	0.99	1.00	1.00	0.99	0.93
8	झ	0.90	0.93	0.89	0.93	0.98	0.96	0.95	0.98	0.99	0.97	0.99	0.98	0.98	0.99	0.99	0.96
9	ञ	0.87	0.85	0.81	0.81	0.94	0.96	0.97	0.96	0.99	0.86	0.99	0.99	0.99	0.99	0.99	0.93
10	ट	0.86	0.86	0.89	0.88	0.96	0.95	0.91	0.93	0.94	0.94	1.00	0.98	0.99	0.99	0.99	0.94
11	ठ	0.78	0.90	0.94	0.89	0.94	0.97	0.92	1.00	0.98	1.00	1.00	1.00	0.99	0.99	1.00	0.95
12	ड	0.78	0.73	0.82	0.84	0.97	0.90	0.91	0.91	0.96	0.91	0.95	0.97	0.97	0.98	0.98	0.91
13	ढ	0.81	0.76	0.86	0.81	0.93	0.92	0.95	0.96	0.90	0.93	0.91	0.97	0.93	0.93	0.97	0.90
14	ण	0.83	0.96	0.89	0.89	0.86	0.96	0.96	0.99	0.97	0.98	0.99	1.00	1.00	1.00	1.00	0.95
15	त	0.82	0.89	0.89	0.74	0.96	0.91	0.90	0.96	0.94	0.94	0.98	0.99	0.97	0.99	0.99	0.92
16	थ	0.66	0.73	0.79	0.72	0.98	0.90	0.86	0.90	0.83	0.91	0.99	0.97	0.93	0.95	0.97	0.87
17	द	0.66	0.80	0.70	0.79	0.88	0.89	0.84	0.90	0.79	0.90	0.96	0.94	0.96	0.99	0.97	0.86
18	ध	0.74	0.76	0.87	0.77	0.88	0.93	0.90	0.92	0.93	0.86	0.97	0.91	0.92	0.94	0.95	0.88
19	न	0.75	0.81	0.87	0.80	0.92	0.94	0.87	0.98	0.88	0.90	0.97	0.99	0.97	0.96	1.00	0.91
20	प	0.57	0.85	0.89	0.73	0.94	0.91	0.88	0.93	0.96	0.97	1.00	0.96	0.98	1.00	0.99	0.90
21	फ	0.88	0.97	0.92	0.85	0.99	0.97	0.95	0.98	0.96	0.97	1.00	1.00	0.98	0.98	1.00	0.96
22	ब	0.68	0.78	0.79	0.82	0.91	0.88	0.96	0.91	0.84	0.94	0.92	0.97	0.97	0.98	0.98	0.89
23	भ	0.62	0.66	0.67	0.73	0.85	0.85	0.80	0.87	0.88	0.83	0.93	0.93	0.95	0.95	0.96	0.83
24	म	0.77	0.71	0.87	0.94	0.84	0.87	0.91	0.88	1.00	0.93	0.94	0.96	0.89	0.97	0.98	0.90
25	य	0.63	0.81	0.82	0.77	0.92	0.89	0.93	0.97	0.85	0.86	0.89	0.95	0.92	0.94	0.94	0.87
26	र	0.82	0.86	0.97	0.82	0.98	0.95	0.94	0.97	0.96	0.98	1.00	0.99	0.99	0.99	0.99	0.95
27	ल	0.84	0.94	0.92	0.84	0.99	0.96	0.95	1.00	1.00	0.98	1.00	1.00	1.00	1.00	0.99	0.96
28	व	0.64	0.73	0.76	0.73	0.85	0.85	0.81	0.91	0.90	0.91	0.97	0.93	0.90	0.98	0.97	0.86
29	श	0.79	0.91	0.79	0.87	0.92	0.90	0.91	0.96	0.90	0.97	0.97	1.00	0.96	1.00	0.99	0.92
30	ष	0.69	0.86	0.86	0.89	0.96	0.96	0.96	0.95	0.98	0.92	0.99	1.00	0.98	1.00	0.99	0.93
31	स	0.70	0.75	0.74	0.69	0.88	0.89	0.85	0.88	0.88	0.93	0.97	0.95	0.95	0.94	0.97	0.86
32	ह	0.73	0.85	0.75	0.74	0.84	0.87	0.89	0.96	0.89	0.88	0.94	0.92	0.93	0.94	0.97	0.87
33	क्ष	0.80	0.86	0.78	0.78	0.97	0.92	0.85	0.93	0.93	0.93	1.00	0.96	1.00	0.99	0.99	0.91
34	त्र	0.72	0.87	0.78	0.83	0.93	0.95	0.90	0.95	0.98	0.97	0.99	0.99	0.99	0.97	0.99	0.92
35	ज्ञ	0.88	0.76	0.81	0.68	0.97	0.95	0.95	0.95	0.94	0.92	0.98	0.96	1.00	0.97	0.99	0.91

Table 11. Character-wise recall produced by an SVM classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
0	क	0.88	0.90	0.89	0.82	1.00	0.94	0.99	0.96	0.97	0.95	0.99	0.98	0.98	0.98	1.00	0.95
1	ख	0.82	0.91	0.87	0.85	0.96	0.94	0.92	0.96	0.99	0.99	1.00	1.00	1.00	0.99	0.99	0.95
2	ग	0.79	0.89	0.84	0.86	0.99	0.93	0.97	1.00	0.95	0.95	0.99	0.99	1.00	1.00	0.99	0.94
3	घ	0.68	0.77	0.80	0.67	0.85	0.96	0.86	0.91	0.96	0.82	0.91	0.91	0.91	0.96	0.94	0.86
4	ङ	0.70	0.82	0.89	0.77	0.94	0.89	0.89	0.96	0.95	0.96	0.96	0.98	0.98	0.97	0.98	0.91
5	च	0.91	0.89	0.95	0.89	0.98	0.94	0.96	1.00	0.94	0.96	0.98	0.99	0.97	0.99	1.00	0.96
6	छ	0.70	0.78	0.81	0.80	0.92	0.91	0.97	0.99	0.94	0.86	0.95	0.94	0.94	0.95	0.97	0.90
7	ज	0.90	0.86	0.82	0.89	0.95	0.97	0.97	0.95	0.99	0.87	0.99	0.97	1.00	0.97	1.00	0.94
8	झ	0.86	0.94	0.93	0.90	0.95	0.97	0.96	1.00	0.97	0.99	1.00	1.00	1.00	1.00	1.00	0.96
9	ञ	0.78	0.93	0.89	0.86	0.96	0.97	0.93	0.91	0.93	0.93	0.99	0.97	0.99	0.99	0.99	0.93
10	ट	0.90	0.94	0.93	0.73	0.94	0.97	0.99	0.99	0.93	0.98	0.97	1.00	0.97	1.00	0.99	0.95
11	ठ	0.80	0.87	0.97	0.98	0.97	0.95	0.92	0.97	0.98	1.00	0.99	0.99	1.00	0.97	1.00	0.96
12	ड	0.81	0.83	0.81	0.83	0.86	0.88	0.91	0.92	0.94	0.89	0.98	1.00	0.98	0.99	0.99	0.91
13	ढ	0.82	0.78	0.81	0.76	0.92	0.93	0.90	0.93	0.89	0.90	0.95	0.97	0.97	0.99	0.98	0.90
14	ण	0.67	0.90	0.90	0.89	0.97	0.93	0.95	0.99	1.00	1.00	0.99	1.00	0.96	0.95	1.00	0.94
15	त	0.95	0.92	0.86	0.83	0.96	0.96	0.95	0.99	0.94	0.97	0.99	0.99	0.97	0.98	0.99	0.95
16	थ	0.76	0.77	0.75	0.73	0.94	0.89	0.92	0.92	0.84	0.85	0.98	0.96	0.91	0.98	0.96	0.88
17	द	0.82	0.76	0.76	0.72	0.91	0.88	0.90	0.92	0.84	0.88	0.91	0.89	0.90	0.91	0.95	0.86
18	ध	0.60	0.84	0.78	0.73	0.91	0.84	0.83	0.94	0.89	0.88	0.96	0.93	0.95	0.96	0.96	0.87
19	न	0.79	0.87	0.81	0.75	0.92	0.96	0.91	0.95	0.92	0.95	0.97	1.00	0.97	0.99	0.99	0.92
20	प	0.63	0.90	0.90	0.82	0.86	0.94	0.88	0.93	0.89	0.96	0.98	1.00	0.95	0.98	0.98	0.91
21	फ	0.86	0.91	0.86	0.88	0.99	0.97	0.97	1.00	0.96	0.95	1.00	0.99	1.00	1.00	0.99	0.96
22	ब	0.59	0.68	0.81	0.78	0.87	0.89	0.93	0.88	0.89	0.91	0.94	0.95	0.91	0.97	0.97	0.86
23	भ	0.72	0.74	0.68	0.74	0.88	0.86	0.89	0.93	0.87	0.92	0.98	0.95	0.94	0.97	0.97	0.87
24	म	0.72	0.69	0.82	0.87	0.87	0.86	0.87	0.84	0.94	0.94	0.95	0.96	0.98	0.96	0.98	0.88
25	य	0.51	0.76	0.78	0.69	0.81	0.86	0.87	0.87	0.86	0.88	0.95	0.96	0.98	1.00	0.96	0.85
26	र	0.69	0.83	0.90	0.87	0.91	0.94	0.95	1.00	0.97	0.99	1.00	0.96	0.97	1.00	0.99	0.93
27	ल	0.87	0.88	0.92	0.87	0.98	0.97	0.91	0.97	0.93	0.99	0.98	0.99	0.97	1.00	1.00	0.95
28	व	0.63	0.70	0.83	0.74	0.79	0.85	0.79	0.94	0.85	0.89	0.86	0.95	0.92	0.96	0.96	0.84
29	श	0.83	0.80	0.76	0.84	0.94	0.97	0.92	0.87	0.80	0.94	0.97	0.95	0.99	0.97	0.99	0.90
30	ष	0.68	0.81	0.83	0.79	0.92	0.88	0.90	0.95	0.94	0.95	0.96	0.95	0.93	0.99	0.99	0.90
31	स	0.59	0.70	0.66	0.71	0.89	0.84	0.75	0.88	0.87	0.87	0.97	0.96	0.94	0.93	0.97	0.84
32	ह	0.74	0.76	0.72	0.66	0.90	0.91	0.75	0.85	0.87	0.81	0.96	0.95	0.95	0.99	0.97	0.85
33	क्ष	0.83	0.72	0.87	0.67	0.93	0.91	0.84	0.92	0.87	0.91	0.99	0.97	0.96	0.96	0.99	0.89
34	त्र	0.64	0.79	0.80	0.83	0.93	0.91	0.92	0.93	0.98	0.94	0.97	0.92	0.97	0.97	0.99	0.90
35	ज्ञ	0.76	0.74	0.69	0.65	0.91	0.93	0.88	0.90	0.89	0.89	0.99	0.97	0.99	0.97	0.99	0.88

Table 12. Character-wise F1 score produced by an SVM classifier for the proposed datasets.

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg
0	क	0.86	0.88	0.89	0.79	0.99	0.93	0.97	0.97	0.94	0.96	0.99	0.97	0.99	0.99	1.00	0.94
1	ख	0.75	0.85	0.85	0.82	0.94	0.92	0.92	0.95	0.95	0.97	0.99	0.99	0.98	0.99	0.99	0.92
2	ग	0.78	0.87	0.82	0.87	0.91	0.93	0.96	0.99	0.96	0.97	0.98	0.99	0.99	0.97	0.99	0.93
3	घ	0.60	0.75	0.72	0.58	0.84	0.87	0.81	0.87	0.86	0.80	0.92	0.88	0.90	0.96	0.94	0.82
4	ङ	0.70	0.79	0.86	0.71	0.89	0.89	0.86	0.93	0.92	0.92	0.97	0.97	0.98	0.96	0.98	0.89
5	च	0.89	0.91	0.91	0.86	0.97	0.96	0.97	0.98	0.95	0.93	0.98	0.99	0.98	0.99	1.00	0.95
6	छ	0.76	0.77	0.80	0.81	0.92	0.92	0.94	0.95	0.94	0.85	0.94	0.95	0.94	0.97	0.96	0.89
7	ज	0.87	0.84	0.84	0.85	0.95	0.98	0.96	0.94	0.94	0.90	0.98	0.98	1.00	0.99	0.99	0.93
8	झ	0.88	0.94	0.91	0.92	0.96	0.97	0.96	0.99	0.98	0.98	0.99	0.99	0.99	0.99	1.00	0.96
9	ञ	0.83	0.89	0.85	0.83	0.95	0.96	0.95	0.93	0.96	0.89	0.99	0.98	0.99	0.99	0.99	0.93
10	ट	0.88	0.90	0.91	0.80	0.95	0.96	0.95	0.96	0.93	0.96	0.98	0.99	0.98	0.99	0.99	0.94
11	ठ	0.79	0.89	0.95	0.93	0.95	0.96	0.92	0.98	0.98	1.00	0.99	0.99	0.99	0.98	1.00	0.95
12	ड	0.79	0.78	0.81	0.83	0.91	0.89	0.91	0.91	0.95	0.90	0.96	0.98	0.97	0.98	0.98	0.90
13	ढ	0.82	0.77	0.83	0.79	0.92	0.93	0.93	0.95	0.89	0.91	0.93	0.97	0.95	0.96	0.98	0.90
14	ण	0.74	0.93	0.90	0.89	0.91	0.95	0.96	0.99	0.98	0.99	0.99	1.00	0.98	0.97	1.00	0.95
15	त	0.88	0.91	0.87	0.78	0.96	0.94	0.92	0.98	0.94	0.96	0.98	0.99	0.97	0.98	0.99	0.94

Table 12. *Cont.*

Cls. No.	Dev. Char	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	Avg.
16	थ	0.70	0.75	0.77	0.73	0.96	0.90	0.89	0.91	0.84	0.88	0.98	0.96	0.92	0.97	0.97	0.88
17	द	0.73	0.78	0.73	0.75	0.89	0.89	0.87	0.91	0.81	0.89	0.93	0.91	0.93	0.95	0.96	0.86
18	ध	0.67	0.80	0.82	0.75	0.90	0.88	0.86	0.93	0.91	0.87	0.97	0.92	0.94	0.95	0.96	0.88
19	न	0.77	0.84	0.84	0.77	0.92	0.95	0.89	0.96	0.90	0.92	0.97	0.99	0.97	0.98	0.99	0.91
20	प	0.60	0.88	0.90	0.78	0.90	0.92	0.88	0.93	0.92	0.97	0.99	0.98	0.96	0.99	0.99	0.91
21	फ	0.87	0.94	0.89	0.87	0.99	0.97	0.96	0.99	0.96	0.96	1.00	0.99	0.99	0.99	1.00	0.96
22	ब	0.63	0.72	0.80	0.80	0.89	0.89	0.95	0.90	0.86	0.92	0.93	0.96	0.94	0.98	0.97	0.88
23	भ	0.66	0.69	0.68	0.73	0.87	0.85	0.84	0.90	0.87	0.87	0.95	0.94	0.94	0.96	0.97	0.85
24	म	0.75	0.70	0.85	0.90	0.85	0.87	0.89	0.86	0.97	0.93	0.94	0.96	0.94	0.97	0.98	0.89
25	य	0.56	0.78	0.80	0.73	0.86	0.87	0.90	0.92	0.86	0.87	0.92	0.95	0.95	0.97	0.95	0.86
26	र	0.75	0.84	0.94	0.85	0.94	0.95	0.95	0.99	0.96	0.98	1.00	0.97	0.98	0.99	0.99	0.94
27	ल	0.86	0.91	0.92	0.85	0.98	0.96	0.93	0.99	0.96	0.98	0.99	0.99	0.98	1.00	1.00	0.95
28	व	0.63	0.72	0.80	0.73	0.82	0.85	0.80	0.93	0.88	0.90	0.91	0.94	0.91	0.97	0.97	0.85
29	श	0.81	0.85	0.77	0.85	0.93	0.93	0.92	0.91	0.85	0.95	0.97	0.97	0.97	0.99	0.99	0.91
30	ष	0.68	0.84	0.84	0.84	0.94	0.92	0.93	0.95	0.96	0.94	0.97	0.98	0.95	0.99	0.99	0.91
31	स	0.64	0.72	0.70	0.70	0.89	0.87	0.79	0.88	0.88	0.90	0.97	0.95	0.95	0.94	0.97	0.85
32	ह	0.74	0.81	0.74	0.70	0.87	0.89	0.81	0.90	0.88	0.84	0.95	0.94	0.94	0.96	0.97	0.86
33	क्ष	0.81	0.79	0.82	0.72	0.95	0.92	0.85	0.93	0.90	0.92	0.99	0.97	0.98	0.97	0.99	0.90
34	त्र	0.68	0.83	0.79	0.83	0.93	0.93	0.91	0.94	0.98	0.96	0.98	0.95	0.98	0.97	0.99	0.91
35	ज्ञ	0.81	0.75	0.75	0.67	0.94	0.94	0.91	0.92	0.92	0.90	0.98	0.97	0.99	0.97	0.99	0.89

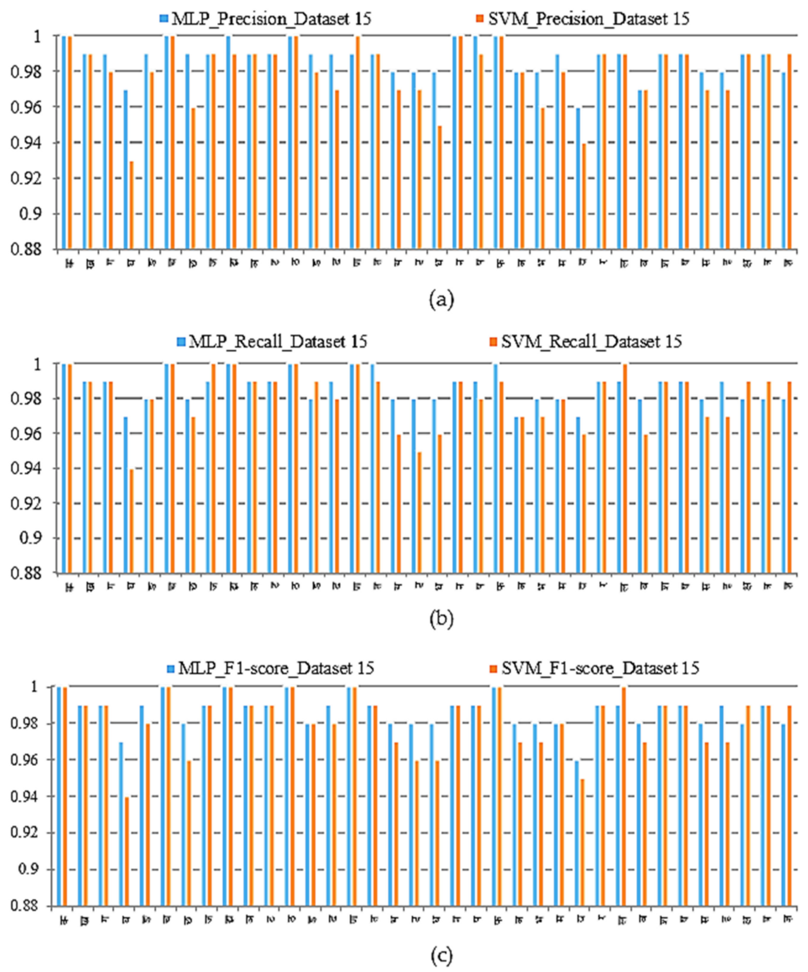

Figure 8. Character-wise comparative analysis of the two classifiers: (**a**) precision analysis (**b**) recall analysis and (**c**) F1-score analysis.

To determine the correct and incorrect predictions for individual character classes, the confusion matrix was generated by the two classifiers for Dataset 15. These are shown in Figure 9.

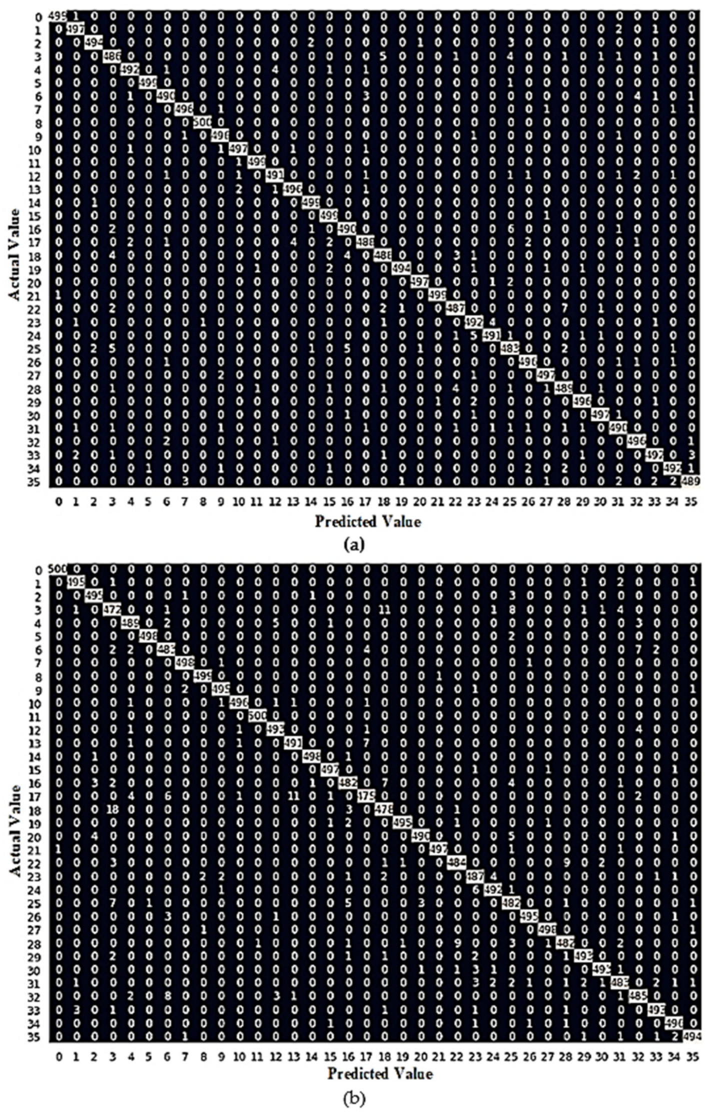

Figure 9. Confusion matrix generated for Dataset 15 by: (**a**) MLP classifier; (**b**) SVM classifier.

4.1.1. Feature Visualization

The Kernel Density Estimate (KDE) is a useful technique to visualize the distribution of observations in the given dataset. It is equivalent to a histogram with the difference of using a continuous probability density curve with one or more dimensions. The proposed scheme employed the KDE plot to visualize the separation of the features related to different character classes. For this purpose, the features were transformed from multi-dimensions to two-dimensions with the help of the PCA method. Figure 10a was plotted with the features received directly from images of the handwritten character dataset, while Figure 10b was plotted with fusion-based hybrid features of Dataset 15. A clear distinction can be made between these two. In Figure 10a, the separation of various character-classes in the feature space was so poor that some of the classes were completely overlapped by the others, and were either poorly or not visible; the examples were the classes 4(ङ), 5(च), 13(ढ), 14(ण), 15(त), and 23(भ). On the contrary, Figure 10b shows a promising separation between all the character classes, except 28(व) and 32(ह). Figure 10 justifies the effectiveness of features collected from the fusion-based approach.

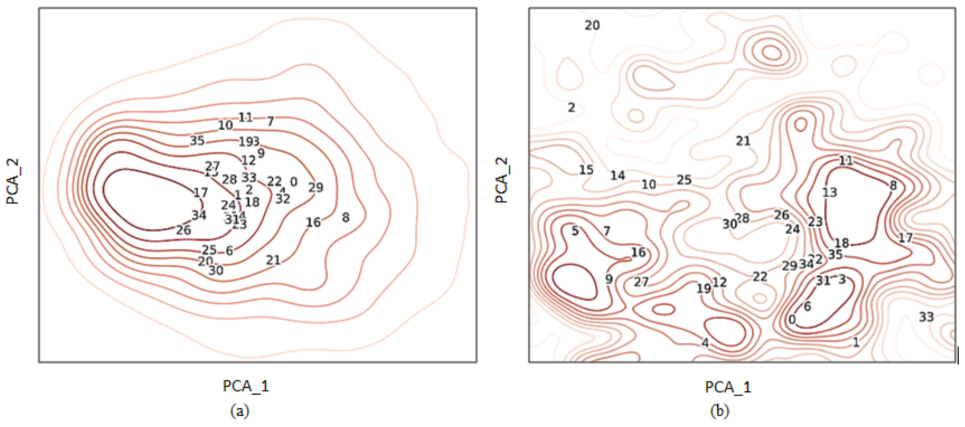

Figure 10. KDE plot for visualizing the separation of features received from (**a**) character images (**b**) fusion-based approach (Dataset 15).

4.1.2. Estimation of Computational Time

Various timings were estimated in relation to feature-extraction and classification for the top performing dataset (i.e., Dataset 15). The mean timings per character w.r.t. feature extraction and dimensionality reduction are compiled in Table 13.

Table 13. Timing details related to the construction of feature-vectors.

Sl. No.	Feature Type	Feature-Extraction Time (ms)	PCA Computation Time (ms)	Total Time (ms)
1	Bior-1.3	2.28	0.15	2.43
2	VGG-19	0.10	0.69	0.79
3	ResNet-50	0.08	0.40	0.48
4	InceptionNet-V3	0.17	0.40	0.57
		Net time		4.27

Referring to Table 13, the net time required was 4.27 msec. for the construction of a hybrid feature-vector from an input image. The timings related to character-classification through a feature-vector are summarized in Table 14.

Table 14. Timing details related to character classification.

Dataset Type	Mean Recognition-Time Per Character (Msec.)	
	MLP Classifier	SVM Classifier
Dataset 15	6.38	17.27

Referring to Tables 13 and 14, the proposed model took total mean time of 10.65 ms and 21.54 msec. in recognition of a test character from the image using MLP and SVM classifiers, respectively.

4.2. Discussions

4.2.1. Feature-Wise Discussion

This section covers a detailed discussion on the strength of features related to various datasets, as mentioned in Table 6.

Referring to Table 6, it can be observed that in the individual category (Dataset 1 to 4), the features received from VGG-19Net (Dataset 2) and ResNet-50 (Dataset 3) produced a comparatively good result over the rest of the features, using MLP and SVM classifiers. The features collected from VGG-19Net were able to produce recognition accuracy of 86.85% on an MLP classifier, and the features received from ResNet-50 were produced 82.66% recognition accuracy on an SVM classifier. These scores were highest in the category.

In the hybrid category consisting of two types of features (Datasets 5 to 10), the combination of features received from VGG-19Net and ResNet-50 (Dataset 8) outperformed the rest of the combinations. This combination produced the best results over both the classifiers in this category; the MLP and SVM classifiers produced 95.37% and 93.92% recognition accuracy, respectively, with Dataset 8.

In the hybrid category with three types of features (Datasets 11 to 14), the fusion of features received from VGG-19Net, ResNet-50, and InceptionV3-Net (Dataset 14) had an upper edge over the remaining combinations in the category. The features of Dataset 14 were able to produce recognition accuracy of 98.03% and 97.51% over the MLP and SVM classifiers, respectively.

The state-of-the-art combination of all the four types of features (Dataset 15) produced excellent results over both the classifiers. With the features of Dataset 15, the MLP and SVM classifiers achieved their best results as 98.73% and 98.18%, respectively. The feature's combination of Dataset 15 also justified the choice of wavelet-based features in association with DCNN-based features, as we could see a significant improvement in the results of both the classifiers when moved from Dataset 14 to Dataset 15.

Dataset 15 has a feature-vector size of 160, which is 84.37% less than that of the character image (i.e., $32 \times 32 = 1024$). This, in turn, reduced the recognition cost of the classifiers significantly.

4.2.2. Classifier-Wise Discussion

Referring to Table 6, Datasets 11, 12, 13, 14 and 15 responded well to both the classifiers and the results were almost comparable. It can be observed that all the datasets produced slightly higher results on the MLP classifier w.r.t. the SVM classifier. This shows the slight upper hand of the non-linear functional approach of the MLP network over the linear separation approach of SVM to solved the proposed multi-class classification problem.

4.2.3. Character-Wise Discussion

Since Dataset 15 was the best performer, we considered it for the character-wise analysis of the model.

Observation of Dataset 15 in Tables 7, 9 and 12 reveals that the MLP classifier recognized the Devnagari characters क (Ka), च (Cha), झ (Jha), ठ (Thha), ण (Adna), and फ (Pha) excellently, as the precision, recall, and F1 score values for these characters are almost equal to 1 (100%); this might be due to the uniqueness of their shapes. For the same reason, the SVM classifier produced fine results for the letters क (ka), च (Cha), झ (Jha), ठ (Thha), ण (Adna), फ (Pha), and ल (La) with respective precision, recall, and F1 score values as 1. It is interesting to know that both the classifiers responded in an excellent manner to the characters क (Ka), च (Cha), झ (Jha), ठ (Thha), ण (Adna), and फ (Pha). This gives a clear indication of uniqueness in the shape of these characters.

The MLP classifier scored satisfactory but had a comparatively low F1 score (refer to Table 9) for the character class य (Ya). The same is true with the SVM classifier (refer to Table 12) for character classes घ (Gha), छ (Chha), द (Da), ध (Dhha), and य (Ya). This might be due to the resemblance of their shapes with some other characters of the complete dataset.

For precise result-analysis of the proposed model, the Results section incorporated individual character-class-wise scores of various performance metrics for the proposed datasets and the proposed classifiers (refer to Tables 7 and 9–12). To relate different values of precision, recall, and F1 score with their practical aspect, let us take one example.

Referring to Table 7, for Dataset 15, the precision value of character य is 0.96 i.e., 96%, which shows that 96% of the total predictions made for the character were correct and for the remaining 4% prediction, the other characters were falsely classified as character य (Ya).

This can also be verified from the confusion matrix given in Figure 8a; on the axis of “Predicted value”, the total 503 predictions can be observed in the column of character य (class no. 25), out of which 483 were correctly predicted as य; and some other characters such as ग (class no. 2), घ (class no. 3), च (class no. 5), ड (class no. 12), थ (class no. 16), प (class no. 20), म (class no. 24), व (class no. 28), and स (class no. 31) were falsely classified as character य for 3, 4, 1, 1, 6, 2, 1, 1, and 1 times, respectively, i.e., total 20 false-positive predictions. It is interesting to know that most of the false predictions were made by character थ (class no. 16), which has a shape resembling the character under test i.e., character य (Ya). The higher the precision value, the lower were the false-positive predictions.

Referring to Table 12, for Dataset 15, the recall value of character य is 0.97 i.e., 97%, which shows that 97% of the total test samples of the character were correctly predicted and for the remaining 3% samples, the character य was falsely classified as some other character. It can be verified from the confusion matrix given in Figure 8a on the axis of “Actual value” that the total 500 samples can be observed in the row of character य (class no. 25), out of which 483 samples were correctly classified as य; and some of the samples of character य were falsely classified as ग (class no. 2), घ (class no. 3), ण (class no. 14), थ (class no. 16), प (class no. 20), व (class no. 28), and त्र (class no. 34) for 2, 5, 1, 5, 1, 2, and 1 times, respectively, i.e., total 17 false-negative predictions. The higher the recall value, the lower were the false-negative predictions.

Referring to Table 9, for Dataset 15, the F1 score value of character य is 0.96 i.e., 96%, which represents a single score for corresponding precision and recall values (refer to Equation (27)). A higher value of the F1 score ensures lower false-positive and false-negative predictions, which is desirable.

Referring to Tables 9 and 12, the cells with the highest F1 score were highlighted for each character class to see the highly efficient dataset in recognition of the given character. For example, Table 12 shows that Datasets 11, 12, 14, and 15 responded excellently to the SVM classifier in recognition of character ख.

Similarly, we can analyse the precision, recall, and F1 score values produced by the MLP (Tables 7, 9 and 12, resp.) and SVM (Tables 10–12, resp.) classifiers for various character classes associated with different datasets.

The average value of precision, recall, and F1 score were estimated on the basis of all the 15 datasets. The top 10 and bottom 10 entries w.r.t. F1 score value (refer to Tables 9 and 12) are summarized in Tables 15 and 16. The characters are arranged in descending order of their mean F1 score.

Table 15. Mean F1 score-wise top 10 characters, including all the 15 datasets.

Classifier	Characters
MLP	झ(Jha), ठ(Thha), च(Cha), ण(Adna), फ(Pha), क(Ka), ट(Taa), त(Ta), ल(La), ञ(Yna).
SVM	झ(Jha), फ(Pha), ठ(Thha), ल(La), च(Cha), ण(Adna), ट(Taa), क(Ka), र(Ra), त(Ta).

Table 16. Mean F1 score-wise bottom 10 characters, including all the 15 datasets.

Classifier	Characters
MLP	थ(Tha), ध(Dha), ब(Ba), य(Ya), भ(Bha), द(Da), घ(Gha), स(Sa), ह(Ha), व(Wa).
SVM	ब(Ba), थ(Tha), ध(Dha), ह(Ha), द(Da), य(Ya), व(Wa), स(Sa), भ(Bha), घ(Gha).

The list of the bottom 10 characters recognized by both the classifiers is the same. This list has a number of resembling character shapes, such as घ(Gha)—ध(Dha), ब(Ba)—व(Wa), and थ(Tha)—य(Ya)—स(Sa). This might be the reason for their comparatively low mean scores.

In feature visualization, although the KDE plots in Figure 10 were drawn with two dimensions only, instead of the total 160 dimensions in Dataset 15, the provision was successful in visualizing the usefulness of the proposed hybrid-features scheme.

5. Conclusions

The proposed scheme provided a comprehensive analysis of the results obtained from individual-based and hybrid-based approaches of feature-extraction using Wavelets, VGG-19Net, ResNet-50, and InceptioNet-V3. The features were examined with two classifiers, namely MLP and SVM. The scheme applies the benefits of deep convolutional neural networks with low cost of recognition. The feature-vector size was successfully reduced by 84.37% with respect to the character image. The scheme managed to score a maximum of 98.73% recognition accuracy with mean character recognition time of 10.65 ms. Table 17 summarizes some excellent work accomplished earlier in the field of handwritten Hindi characters recognition; it includes state-of-the-art models related to the feature-based classification. Here, it is reasonable to mention that no standard dataset of handwritten Hindi script is available in the public domain, restricting a comparison of related models on a common platform [3]. However, one can gain some idea about the performance of the proposed work by comparing it to existing methods.

Table 17. Summary of previous work done in the field with the proposed one.

Scheme 1.	Particular	Features Used	Classif. Scheme	No. of Test Samples	No. of Features for Classif.	Max. Recogn. Test-Acc. (%)	Data Set Used
1	Arora et al. 2008 [57]	Line fitting, intersection-points, shadow and chain code-based features	MLP	1568	296	92.8	[57]
2	Satish Kumar [58]	Gradient, neighborhood pixels weight, and distance transform-based features	SVM	25,000	192	94.3	[58]
3	Singh et al. 2011 [59]	Curvelet transform-based features	KNN	7965	190	93.8	[59]
4	Dixit et al. 2014 [27].	Wavelet transform-based features	MLP	600	256	70	[27]
5	Jangid et al. 2014 [60]	Local auto-correlation of gradient	SVM	NA	612	95.21	[61]
6	Khanduja et al. 2015 [15].	Hybrid of structural and statistical features	MLP	4000	462	91.4	[61]
7	Singh et al. 2015 [28].	Chain code, zone-based centroid, background directional distribution and distance profile features	MLP and SVM	4000	625	97.61 (SVM)	[28]
8	Jangid et al. 2016 [17].	Fisher discriminant-based features	SVM	10,850	256	96.58	[62]
9	Sarkhel et al. 2017 [30].	CNN-based features	SVM	3894	4096, 2560 & 1792	95.180	[63]
10	Nikita Singh, 2018 [16]	Histogram of gradients	MLP	900	1224	97.5	[16]
11	Gupta et al. 2019 [29]	HOG, convex hull, and longest run-based features	SVM	10,800	5568	94.15	[55]
12	Yadav et al. 2017 [64]	Projection profile-based features with HOG	Quadratic SVM	4428	NA	96.6	[62]
13	Proposed scheme	Hybrid of BDWT and DCNN-based features	MLP and SVM	18,000	160	98.73 (MLP)	[55]

The proposed model was experimented with the fresh approach of feature-extraction; for the very first time, the features received from pre-trained DCNN models were combined with the handcrafted features obtained from BDWT. The outcomes are promising, as the proposed scheme gained comparable results with respect to the mentioned benchmarking models, even with the smallest size of the feature-vector. The model achieved percentage reduction in feature-vector size in the range of 15.78 to 97.12, with respect to the given benchmarking models (refer to Table 17). The classification cost of a machine learning algorithm is directly dependent on the size of the feature-vector. In this regard, the proposed model was successful in reducing the classification cost. The features were so effective that the classification error was limited to 1.27% only. The presented work will be helpful for further development of recognition models related to handwritten Hindi characters.

In future, the proposed scheme could be extended by including some more advanced pre-trained deep convolutional networks like InceptionV4-net, ResNeXt, Squeeze-Excitation network (SE-Net), etc. for feature extraction. Other classification algorithms like KNN, RNN, LSTM, Random-forest, and Naïve-Bayes could also be analyzed for the extended version.

Author Contributions: Conceptualization, D.R.; Data curation, D.R. and O.P.M.; Formal analysis, O.P.M.; Investigation, D.R.; Methodology, O.P.M.; Supervision, A.R.G. and P.S.; Validation, A.R.G.; Writing–original draft, D.R.; Writing–review & editing, A.R.G., H.H.A. and P.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Memon, J.; Sami, M.; Khan, R.A.; Uddin, M. Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). *IEEE Access* **2020**, *8*, 142642–142668. [CrossRef]
2. Yadav, M.; Purwar, R.K.; Mittal, M. Handwritten Hindi character recognition: A review. *IET Image Process.* **2018**, *12*, 1919–1933. [CrossRef]
3. Sharma, R.; Kaushik, B. Offline recognition of handwritten Indic scripts: A state-of-the-art survey and future perspectives. *Comput. Sci. Rev.* **2020**, *38*, 100302. [CrossRef]
4. Jayadevan, R.; Kolhe, S.R.; Patil, P.M.; Pal, U. Offline Recognition of Devanagari Script: A Survey. *IEEE Trans. Syst. MAN Cybern. C Appl. Rev.* **2011**, *41*, 782–796. [CrossRef]
5. Sethi, I.K.; Chatterjee, B. Machine Recognition of Hand-printed Devnagri Numerals. *IETE J. Res.* **1976**, *22*, 532–535. [CrossRef]
6. Bhattacharya, U.; Parui, S.K.; Shaw, B.; Bhattacharya, K. Neural Combination of ANN and HMM for Handwritten Devanagari Numeral Recognition. In Proceedings of the Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule, France, October 2006; Suvisoft: Tampre, Finland, 2006.
7. Arora, S.; Bhattacharjee, D.; Nasipuri, M.; Malik, L. A Two Stage Classification Approach for Handwritten Devanagari Characters. In Proceedings of the International Conference on Computational Intelligence and Multimedia Applications 2007, Sivakasi, India, 13–15 December 2007; IEEE Press: Piscataway, NJ, USA, 2007; pp. 404–408.
8. Bajaj, R.; Dey, L.; Chaudhury, S. Devnagari numeral recognition by combining decision of multiple connectionist classifiers. *Sadhana* **2002**, *27*, 59–72. [CrossRef]
9. Sharma, N.; Pal, U.; Kimura, F.; Pal, S. Recognition of Off-Line Handwritten Devnagari Characters Using Quadratic Classifier N. In *ICVGIP 2006*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4338, pp. 805–816.
10. Pal, U.; Sharma, N.; Wakabayashi, T.; Kimura, F. Off-line handwritten character recognition of devnagari script. In Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brazil, 23–26 September 2007; Volume 1, pp. 496–500. [CrossRef]
11. Deshpande, P.S.; Malik, L.; Arora, S. Fine classification and recognition of hand written Devnagari characters with regular expressions & minimum edit distance method. *J. Comput.* **2008**, *3*, 11–17. [CrossRef]
12. Iamsa-At, S.; Horata, P. Handwritten character recognition using histograms of oriented gradient features in deep learning of artificial neural network. In Proceedings of the International Conference on IT Convergence and Security (ICITCS) 2013, Macao, China, 16–18 December 2013; IEEE Press: Piscataway, NJ, USA, 2013. [CrossRef]
13. Shitole, S.; Jadhav, S. Recognition of Handwritten Devnagari Characters using Linear Discriminant Analysis. In Proceedings of the Second International Conference on Inventive Systems and Control (ICISC 2018), Coimbatore, India, 19–20 January 2018; IEEE Press: Piscataway, NJ, USA, 2018; Volume 1, pp. 100–103.
14. Rojatkar, D.V.; Chinchkhede, K.D.; Sarate, G.G. Design and analysis of LRTB feature based classifier applied to handwritten Devnagari characters: A neural network approach. In Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI 2013), Mysore, India, 22–25 August 2013; IEEE Press: Piscataway, NJ, USA, 2013; pp. 96–101. [CrossRef]
15. Khanduja, D.; Nain, N.; Panwar, S. A hybrid feature extraction algorithm for Devanagari script. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2015**, *15*, 1–10. [CrossRef]
16. Singh, N. An Efficient Approach for Handwritten Devanagari Character Recognition based on Artificial Neural Network. In Proceedings of the 5th International Conference on Signal Processing and Integrated Networks (SPIN 2018), Noida, India, 22–23 February 2018; IEEE Press: Piscataway, NJ, USA, 2018; pp. 894–897. [CrossRef]

17. Jangid, M.; Srivastava, S. Similar handwritten devanagari character recognition by critical region estimation. In Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI 2016), Jaipur, India, 21–24 September 2016; IEEE Press: Piscataway, NJ, USA, 2016; pp. 1936–1939. [CrossRef]
18. Puri, S.; Singh, S.P. An efficient Devanagari character classification in printed and handwritten documents using SVM. *Procedia Comput. Sci.* **2019**, *152*, 111–121. [CrossRef]
19. Sethi, I.K.; Chatterjee, B. Machine recognition of constrained hand printed devanagari. *Pattern Recognit.* **1977**, *9*, 69–75. [CrossRef]
20. Parui, S.K.; Shaw, B. Offline handwritten Devanagari word recognition: An HMM based approach. In *International Conference on Pattern Recognition and Machine Intelligence*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 528–535. [CrossRef]
21. Holambe, A.N.; Holambe, S.N.; Thool, R.C. Comparative study of devanagari handwritten and printed character & numerals recognition using Nearest-Neighbor classifiers. In Proceedings of the 3rd International Conference on Computer Science and Information Technology (ICCSIT 2010), Chengdu, China, 9–11 July 2010; IEEE Press: Piscataway, NJ, USA, 2010; Volume 1, pp. 426–430. [CrossRef]
22. Pal, U.; Wakabayashi, T.; Kimura, F. Comparative study of Devnagari handwritten character recognition using different feature and classifiers. In Proceedings of the 10th International Conference on Document Analysis and Recognition ICDAR, Barcelona, Spain, 26–29 July 2009; IEEE Press: Piscataway, NJ, USA, 2009; pp. 1111–1115. [CrossRef]
23. Hanmandlu, M.; Nath, A.V.; Mishra, A.C.; Madasu, V.K. Fuzzy model based recognition of Handwritten Hindi Numerals using bacterial foraging. In Proceedings of the 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), Melbourne, VIC, Australia, 11–13 July 2007; IEEE Press: Piscataway, NJ, USA, 2007; pp. 309–314. [CrossRef]
24. Gupta, D.; Bag, S. CNN-based multilingual handwritten numeral recognition: A fusion-free approach. *Expert Syst. Appl.* **2021**, *165*, 113784. [CrossRef]
25. Garg, T.; Garg, M.; Mahela, O.P.; Garg, A.R. Convolutional Neural Networks with Transfer Learning for Recognition of COVID-19: A Comparative Study of Different Approaches. *AI* **2020**, *1*, 586–606. [CrossRef]
26. Verma, B.K. Handwritten Hindi Character Recognition using multilayer perceptron and radial basis function neural networks. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; IEEE Press: Piscataway, NJ, USA, 1995; Volume 4, pp. 2111–2115. [CrossRef]
27. Dixit, A.; Navghane, A.; Dandawate, Y. Handwritten Devanagari character recognition using wavelet based feature extraction and classification scheme. In Proceedings of the Annual IEEE India Conference (INDICON 2014), Pune, India, 11–13 December 2014; IEEE Press: Piscataway, NJ, USA, 2015; pp. 1–3. [CrossRef]
28. Singh, A.; Maring, K.A. Handwritten Devanagari Character Recognition using SVM and ANN. *Int. J. Adv. Res. Comput. Commun. Eng.* **2015**, *4*, 123–127. [CrossRef]
29. Gupta, A.; Sarkhel, R.; Das, N.; Kundu, M. Multiobjective optimization for recognition of isolated handwritten Indic scripts. *Pattern Recognit. Lett.* **2019**, *128*, 318–325. [CrossRef]
30. Sarkhel, R.; Das, N.; Das, A.; Kundu, M.; Nasipuri, M. A Multi-scale Deep Quad Tree Based Feature Extraction Method for the Recognition of Isolated Handwritten Characters of popular Indic Scripts. *Pattern Recognit.* **2017**, *71*, 78–93. [CrossRef]
31. Chakraborty, B.; Shaw, B.; Aich, J.; Bhattacharya, U.; Parui, S.K. Does deeper network lead to better accuracy: A case study on handwritten devanagari characters. In Proceedings of the 13th IAPR International Workshop on Document Analysis Systems (DAS 2018), Vienna, Austria, 24–27 April 2018; IEEE Press: Piscataway, NJ, USA, 2018; pp. 411–416. [CrossRef]
32. Jangid, M.; Srivastava, S. Handwritten Devanagari character recognition using layer-wise training of deep convolutional neural networks and adaptive gradient methods. *J. Imaging* **2018**, *4*, 41. [CrossRef]
33. Sonawane, P.K.; Shelke, S. Handwritten Devanagari Character Classification using Deep Learning. In Proceedings of the International Conference on Information, Communication, Engineering and Technology (ICICET 2018), Pune, India, 29–31 August 2018; IEEE Press: Piscataway, NJ, USA, 2018; pp. 1–4. [CrossRef]
34. ImageNet. Available online: https://www.image-net.org/challenges/LSVRC/ (accessed on 4 June 2021).
35. Shelke, S.; Apte, S. A novel multistage classification and Wavelet based kernel generation for handwritten Marathi compound character recognition. In Proceedings of the 2011 International Conference on Communications and Signal Processing, Kerala, India, 10–12 February 2011; IEEE Press: Piscataway, NJ, USA, 2011; pp. 193–197. [CrossRef]
36. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **2021**, *109*, 43–76. [CrossRef]
37. Kandel, I.; Castelli, M. Transfer Learning with Convolutional Neural Networks for Diabetic Retinopathy Image. *J. Appl. Sci.* **2021**, *10*, 2021. [CrossRef]
38. Loey, M.; Mukdad, N.; Hala, Z. Deep Transfer Learning in Diagnosing Leukemia in Blood Cells. *J. Comput.* **2020**, *9*, 29. [CrossRef]
39. Narayanan, B.N.; Hardie, R.C.; Krishnaraja, V.; Karam, C.; Salini, V.; Davuluru, P. Transfer-to-Transfer Learning Approach for Computer Aided Detection of COVID-19 in Chest Radiographs. *AI* **2020**, *1*, 539–557. [CrossRef]
40. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015—Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015; pp. 1–14.
41. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; IEEE Press: Piscataway, NJ, USA, 2016; pp. 2818–2826. [CrossRef]

42. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; IEEE Press: Piscataway, NJ, USA, 2016; pp. 770–778. [CrossRef]
43. Ouahabi, A. *Signal and Image Multiresolution Analysis*; ISTE-Wiley: London, UK; Hoboken, NJ, USA, 2013.
44. Choose a Wavelet—MATLAB & Simulink. Available online: https://www.mathworks.com/help/wavelet/gs/choose-a-wavelet.html (accessed on 9 July 2021).
45. Burrus, C.S.; Gopinath, R.A.; Guo, H. *Introduction to Wavelets and Wavelet Transforms: A Primer*, 1st ed.; Pearson College Div: Upper Saddle River, NJ, USA, 1997.
46. Prakash, O.; Park, C.M.; Khare, A.; Jeon, M.; Gwak, J. Multiscale fusion of multimodal medical images using lifting scheme based biorthogonal wavelet transform. *Optik* **2019**, *182*, 995–1014. [CrossRef]
47. Odegard, J.E.; Sidney Burrus, C. Smooth biorthogonal wavelets for applications in image compression. In Proceedings of the 1996 IEEE Digital Signal Processing Workshop Proceedings, Loen, Norway, 1–4 September 1996; IEEE Press: Piscataway, NJ, USA, 1996; pp. 73–76. [CrossRef]
48. Sweldens, W. The lifting scheme: A construction of second generation wavelets. *Soc. Ind. Appl. Math.* **1998**, *29*, 511–546. [CrossRef]
49. Du, Y.C.; Hu, W.C.; Shyu, L.Y. The effect of data reduction by independent component analysis and principal component analysis in hand motion identification. In Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, San Francisco, CA, USA, 1–5 September 2004; IEEE Press: Piscataway, NJ, USA, 2004; Volume 26, pp. 84–86. [CrossRef]
50. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Umar, A.M.; Linus, O.U.; Arshad, H.; Kazaure, A.A.; Gana, U.; Kiru, M.U. Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition. *IEEE Access* **2019**, *7*, 158820–158846. [CrossRef]
51. Kingma, D.P.; Ba, J.L. ADAM: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015—Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015; pp. 1–15.
52. Zanaty, E.A. Support Vector Machines (SVMs) versus Multilayer Perception (MLP) in data classification. *Egypt. Inform. J.* **2012**, *13*, 177–183. [CrossRef]
53. Mathematical Introduction for SVM and Kernel Functions—Tsmatz. Available online: https://tsmatz.wordpress.com/2020/06/01/svm-and-kernel-functions-mathematics/ (accessed on 6 June 2021).
54. Math Behind SVM(Kernel Trick). This Is PART III of SVM Series | by MLMath.io | Medium. Available online: https://medium.com/@ankitnitjsr13/math-behind-svm-kernel-trick-5a82aa04ab04 (accessed on 6 June 2021).
55. Acharya, S.; Pant, A.K.; Gyawali, P.K. Deep learning based large scale handwritten Devanagari character recognition. In Proceedings of the SKIMA 2015—9th International Conference on Software Knowledge, Information Management and Applications, Kathmandu, Nepal, 15–17 December 2015; IEEE Press: Piscataway, NJ, USA, 2015; pp. 1–6.
56. Cross-Validation in Machine Learning | by Prashant Gupta | Towards Data Science. Available online: https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f (accessed on 7 June 2021).
57. Arora, S.; Bhattacharjee, D.; Nasipuri, M.; Basu, D.K.; Kundu, M. Combining multiple feature extraction techniques for Handwritten Devnagari Character recognition. In Proceedings of the 2008 IEEE Region 10 and the Third international Conference on Industrial and Information Systems, Kharagpur, India, 8–10 December 2008; IEEE Press: Piscataway, NJ, USA, 2008; pp. 1–6. [CrossRef]
58. Kumar, S. Performance Comparison of Features on Devanagari Hand-printed Dataset. *Int. J. Recent Trends Eng.* **2015**, *1*, 33–37.
59. Singh, B. An Evaluation of Different Feature Extractors and Classifiers for Offline Handwritten Devnagari Character Recognition. *J. Pattern Recognit. Res.* **2011**, *2*, 269–277. [CrossRef]
60. Jangid, M.; Srivastava, S. Gradient Local Auto-Correlation for Handwritten Devanagari Character Recognition Mahesh. In Proceedings of the 2014 International Conference on High Performance Computing and Applications (ICHPCA), Bhubaneswar, India, 22–24 December 2014; IEEE Press: Piscataway, NJ, USA, 2014.
61. Dongre, V.J.; Mankar, V.H. Development of Comprehensive Devnagari Numeral and Character Database for Offline Handwritten Character Recognition. *Appl. Comput. Intell. Soft Comput. Hindawi Publ. Corp.* **2012**, *2012*, 871834. [CrossRef]
62. ISI Image Databases of Handwritten Isolated Characters. Available online: https://www.isical.ac.in/~{}ujjwal/download/Devanagaribasiccharacter.html (accessed on 6 June 2021).
63. HPL Handwriting Datasets. Available online: http://lipitk.sourceforge.net/hpl-datasets.htm (accessed on 20 June 2021).
64. Yadav, M.; Purwar, R. Hindi handwritten character recognition using multiple classifiers. In Proceedings of the 7th International Conference on Cloud Computing, Data Science & Engineering (Confluence, 2017), Noida, India, 12–13 January 2017; IEEE Press: Piscataway, NJ, USA, 2017; pp. 149–154. [CrossRef]