Mohammad Idrees Bhat* and B. Sharada

Spectral Graph-based Features for Recognition of Handwritten Characters: A Case Study on Handwritten Devanagari Numerals

https://doi.org/10.1515/jisys-2017-0448
Received August 31, 2017; previously published online July 21, 2018.

Abstract: Interpretation of different writing styles, unconstrained cursiveness and relationship between different primitive parts is an essential and challenging task for recognition of handwritten characters. As feature representation is inadequate, appropriate interpretation/description of handwritten characters seems to be a challenging task. Although existing research in handwritten characters is extensive, it still remains a challenge to get the effective representation of characters in feature space. In this paper, we make an attempt to circumvent these problems by proposing an approach that exploits the robust graph representation and spectral graph embedding concept to characterise and effectively represent handwritten characters, taking into account writing styles, cursiveness and relationships. For corroboration of the efficacy of the proposed method, extensive experiments were carried out on the standard handwritten numeral Computer Vision Pattern Recognition, Unit of Indian Statistical Institute Kolkata dataset. The experimental results demonstrate promising findings, which can be used in future studies.

Keywords: Writing styles, unconstrained cursiveness, primitive relationships, feature representation, graph representation, spectral graph embedding.

1 Introduction

Optical character recognition (OCR) is concerned with automatic recognition of scanned and digitised images of text by a computer. These scanned images of text undergo various manipulations and then encoded with character codes such as American Standard Code for Information Interchange (ASCII), Unicode, and so on. The OCR system tries to bridge the communication gap between man and machine and aides in automation of office with saving of considerable amount of time and human effort. Despite decades of research on different issues related to OCR [15, 26], research on handwritten characters has been less than satisfactory. It is an essential and challenging task for the community of pattern recognition. It is primarily because of the absence of a fixed structure, the presence of numerous character shapes, cursiveness and the difference in inter and intra writer styles. Potential practical applications of it are included in the automatic reading of postal codes, bank cheques, employee id, data entry, zip codes, and so on. Thus, recognition of handwritten characters is still an open area of research. In general, problems are associated with all handwritten documents. In this paper, we consider a case study of handwritten Devanagari numerals, because of its importance in the Indian context.

One important question is how to give adequate representation/description of the underlying object (handwritten character) such that any recognition algorithm can be applied. Representation of an object is done through two ways, namely statistical representation and structural representation. In statistical representation, the character is represented as a feature vector comprising 'n' measurements or values and can be thought of as a point in n-dimensional vector space, that is, $F = (f_1, f_2, \ldots, f_n) \in R^n$. However, it has two representational limitations: first, dimension is fixed a priori, that is, all vectors in a recognition system have

***Corresponding author: Mohammad Idrees Bhat,** Department of Studies in Computer Science, University of Mysore, Manasagangothri, Mysore-570006, Karnataka, India, e-mail: idrees11@yahoo.com
B. Sharada: Department of Studies in Computer Science, University of Mysore, Manasagangothri, Mysore-570006, Karnataka, India

to agree with the same length irrespective of the varying size of the underlying objects, and second, they are inadequate in representing binary relationships that exist in primitive parts of the underlying object. Despite these, they are extensively used because of their flexible and computationally efficient mathematical base. For example, sum, product, mean, and so on, which are basic artefacts for many pattern recognition algorithms, can easily be computed. On the other hand, structural representation is based on a symbolic data structure, namely, graphs. The aforementioned limitations of feature vectors can be circumvented by graph representation [17, 56]. However, little algebraic support (less mathematical flexibility) and computationally expensive nature of many algorithms are major drawbacks to it. Compared to the feature representation method, graphs provide robust representation formalism for the description of two-dimensional nature of handwritten characters, namely, style variance, shape transformations, cursiveness and size variance [56]. In this work, in order to exploit the advantages of both, we give graph representation to handwritten numerals to capture different writing styles, cursiveness and size variability. Afterwards, graphs are transformed into vector space by the concept of spectral graph theory (SGT) to characterise the numeral graphs. The rest of the paper is organised into five sections: Section 2 gives brief literature on the handwritten Devanagari numeral recognition system. An overview of definitions/illustrations of the terminologies used with respect to graph and spectral graph theory is given in Section 3. In Section 4 details about the proposed system are given. The recognition experiment is described in Section 5, starting with a description of the dataset and experimental setup, followed by experimental results and concluded by comparison with related work. Finally, future work and conclusion are drawn in Section 6.

2 Related Works

Over the years, an enormous amount of research work has been carried out in an attempt to make OCR a reality. Different studies have explored various techniques such as template matching [12], multi-pass hybrid method [54], syntactic features [42], shadow-based features [6, 46], gradient features [33, 45] and convolutional neural network based features [35], to name just a few. Robust and stable features that are discriminating in feature space are an indispensable component in any recognition system. Inevitable characteristic of such features is that they should withstand different types of variations (style, size, etc.) and shape transformations, namely, rotation, scale, translation and reflection. Selection and extraction of such features in handwritten characters in the Indian context have been attempted by a number of researchers.

In Ref. [3], moment features (left, right, upper and lower profile curves), descriptive component features and density features are combined for neural network-based architecture for recognition. The main aim of extracting these types of features is to capture different stylistic variations. In Ref. [10], after giving wavelet-based multi-resolution representation, a numeral is subjected to the multi-stage recognition process. In each stage, a distinct multi-layer perceptron classifier is used which either performs recognition or rejection. Thereafter, recognition for a rejected numeral is attempted at the next higher level. A fuzzy model-based system is proposed in Ref. [31]; numerals are represented in the form of the exponential membership function, which behaves as a fuzzy model. Later recognition is performed by modifying exponential membership functions fitted to the fuzzy sets. Fuzzy sets are extracted from features comprising normalised distances using the Box approach. An attempt is made in Ref. [43] to extract moment invariant features based on correlation coefficient, perturbed moments, image partitions and principal component analysis (PCA). These features are then used with the Gaussian distribution function for recognition purpose. In Ref. [41], translation and scale invariance of numerals are achieved by exploiting geometric moments such as Zernike moments. Extensive experiments were carried out on a large dataset that revealed the robustness of the proposed model. After giving graph representation different graph matching techniques are used such as sub-graph isomorphism, maximum common sub-graph and graph edit distance for holistic recognition of Devanagari words [36], Oriya digits [28] and Devanagari numerals [7], respectively. However, the robustness of the graph representation is overshadowed by time complexity in these approaches.

A novel scheme based on edge histogram features is proposed in Ref. [55]; scanned numeral images are pre-processed with splines together with PCA in order to improve the recognition performance. A local-based

approach is proposed in Ref. [5], which exploits 16-segment display concept, extracted from half-toned binary images of numerals. A novel approach for recognising handwritten numerals of five Indian subcontinent scripts is proposed in Ref. [22]. Handwritten numerals are characterised by a combination of features such as PCA/modular PCA (MPCA) and quadtree-based hierarchically derived longest run. The efficacy of the proposed approach is validated by conducting extensive experiments on various datasets, and the results demonstrate significant development in recognition performance. A global-based approach is proposed in Ref. [4], in which features are extracted from end points of numeral images. Thereafter, recognition is carried out with the neuromagnetic model. The feature level fusion-based approach is attempted in Ref. [51], in which global and local features are combined together for artificial neural network-based recognition. Several techniques gained importance due to their performance such as chain code features [49], feature sub-selection [4], Zernike moments [37] and structural features [11]. For a comprehensive survey, we refer readers to [2, 32, 39].

From the literature survey, we observe that many researchers have addressed the problem of handwritten Devanagari numeral recognition by addressing separate objectives (shape transformations, style variations, etc.). However, no attempts were made to address the problem as a whole. As numerals written by people are with different writing styles, even variation of style exists within writer also; handwritten numeral recognition seems to be difficult and challenging. Thus, there is a scope for various attempts in this direction. Also, the reported works clearly indicate that the attempts have been made only by giving feature representation. However, as stated earlier, feature representation implicates two limitations, namely, size constraint and inability to represent binary relationships. These two limitations are severe in representing inherent two-dimensional nature of handwriting. With this observation, if these two limitations can be removed from recognition systems, greater and reliable recognition accuracies can be achieved. Hence, there is a scope to devise a model to circumvent stated limitations by providing robust alternative representation. From such a representation, besides representing object properties, we expect that inherent two-dimensional information is adequately modelled and binary relationships are preserved.

Graph representation models dependencies, binary relations among different primitive parts (by edges), besides describing object properties. Moreover, flexible in representing different object size in an application and invariant to shape transformations (scale, rotation, translation, reflection and mirror image) as well [18]. These characteristics of graphs are extremely beneficial to cope with different writing styles and cursiveness. Also, from the survey, with different applications such as image classification [44], image segmentation [50], synthetic graph classification [47], and many more, we observe that SGT is more effective to characterise the graphs under consideration. SGT is a branch of mathematics that is primarily concerned with describing the structural properties of graphs by extracting eigenvalues of different graph-associated matrices. The eigenvalues form the spectrum of the graph and exhibit interesting properties which can be exploited for recognition purposes. To enhance the recognition performance classifier fusion at the decision level is also used. The Computer Vision Pattern Recognition, Unit of Indian Statistical Institute Kolkata (CVPR Unit, ISI Kolkata) dataset is employed as a dataset due to its popularity, availability and its complexity. Recognition results are lesser than to the best result claimed in [40]. However, the main aim was not to outperform it but to circumvent stated limitations by giving graph representation and observe the results (Figure 1).

Figure 1: Illustration of Numeral Images with Several Intra-class Variations with Respect to Size and Style.

3 Required Graph Terminologies

Brief and concise illustrations are given for various terminologies used in this study vis-à-vis graph theory and SGT. However, for comprehensive reading, we refer readers to [13, 16, 23, 29].

Definition 1 (Graph). A graph is a four-tuple $G = (V, E, \mu, \nu)$, where
- V set of vertices (or nodes); cardinality of it is the order of the graph
- $E \subseteq V \times V$ set of edges; cardinality of it is the size of the graph
- $\mu: V \rightarrow l_v$ *associating labels*, l_v, with each vertex in V
- $\nu: E \rightarrow l_e$ *associating labels*, l_e, *with each edge in* E.

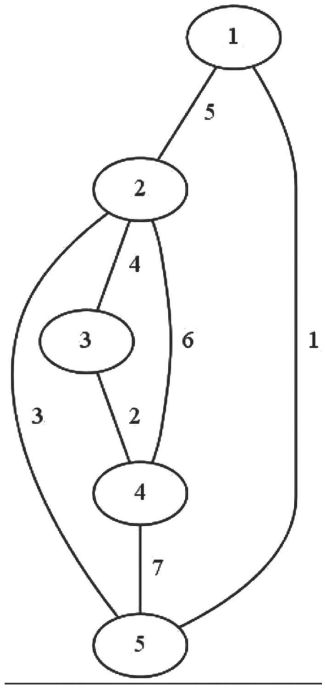
A *directed graph* or *digraph* G is a graph in which all edges e in E are directed from one vertex to another, that is, vertices are ordered pairs in V. An *undirected graph* G is a graph in which all edges e in E are bidirectional, that is, vertices are unordered pairs in V. A *weighted graph* G is a graph in which each edge e in E is assigned a numerical weight by some *weighting function* $w(e_i)$. Mainly non-negative numeric values are used (called the cost of the edges). One such weighting function $w(e_i)$ is the length of the edge e in E. The *degree of a vertex* v denoted by $d(v)$ in G is the total number of vertices that are adjacent to it. There are different matrices associated with graphs which are important such as adjacency matrix and Laplacian matrix. In a graph G with $|V|$ vertices, an *adjacency matrix* ($A(G)$) is a $|V| \times |V|$ matrix. Each a_{ij} in $A(G)$ is 1 if the vertices $\{v_i, v_j\}$ in V are adjacent, otherwise 0. The *Laplacian matrix* ($L(G)$) of graph G is defined as $L(G) = D(G) - A(G)$, where $D(G)$ and $A(G)$ are the *degree and adjacency matrix of graph* G, respectively. Each l_{ij} in $L(G)$ is $\deg(v_i)$ if $\{v_i = v_j\}$ $\forall\, i, j$, -1 if edges e in E are adjacent ($\forall\, i \neq j$) and 0 otherwise. The *weighted adjacency matrix* $WA(G)$ is constructed by removing all entries where $\{v_i, v_j\} = 1$ in $A(G)$ with respective weights assigned by a weighting function $w(\{v_i, v_j\})$. The weighted Laplacian matrix $WL(G) = D(G) - WA(G)$, where $D(G)$ is a degree matrix. Each l_{ij} in $WL(G)$ is defined as: $\deg(v_i)$ if $i = j$, negative times weight assigned by $w(e_i)$ to edges in $WA(G)$ and 0 otherwise. The *distance matrix* $Dist(G)$ of vertices in a graph G is the $|V| \times \mid V \mid$ matrix, which contains pairwise distances (provided by a weighting function, $w(e_i)$) between each v in V, that is, distances are included even for non-adjacent nodes v in V. Despite robust structural representational formalism of objects, as stated earlier, graph-based methods in pattern recognition (like *graph matching*) have major limitations. These limitations are computationally expensive nature of algorithms and the presence of little algebraic properties (basic operations required in many pattern recognition algorithms such as sum, mean, and product are not defined in a standard way). In order to overcome these limitations, graphs are transformed into low-dimensional vector space; such a technique is called *graph embedding* $\varphi: G \rightarrow R^n$. One such technique is *spectral graph embedding* (SGE), in which graphs are transformed into vector space by the *spectrum of the graph*. The spectrum of graph G (where G can be represented by any graph-associated matrix M, in this study $WA(G)$, $WL(G)$ and $Dist(G)$) is the set of *eigenvalues*, together with their algebraic multiplicities (number of times they occur). Representation of any graph-associated matrix in terms of its *eigenvalues* and *eigenvectors* is called its *eigendecomposition/spectral decomposition*. For better illustration, let $G(5, 7)$ be the graph in which each edge e is *weighted (labelled)* arbitrarily, and then the desired matrices can be extracted, as shown in Figure 2. It should be noted that there is a subtle difference between the label and weight of the graph; in this study label and weight refer to the same and are used interchangeably.

4 Proposed Model

Various steps involved in the proposed handwritten Devanagari numeral recognition model are shown in Figure 3. These steps are explained in the following subsections.

4.1 Image Pre-processing

Image pre-processing deals with reducing variations on scanned images of handwritten numerals caused by noise. In this study, scanned numeral images are first filtered by difference of Gaussian filtering, then



$$WA(G) = \begin{bmatrix} 0 & 5 & 0 & 0 & 1 \\ 5 & 0 & 4 & 6 & 3 \\ 0 & 4 & 0 & 2 & 0 \\ 0 & 6 & 2 & 0 & 7 \\ 1 & 3 & 0 & 7 & 0 \end{bmatrix} \quad D(G) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix} \quad WL(G) = \begin{bmatrix} 2 & -5 & 0 & 0 & -1 \\ -5 & 3 & -4 & -6 & -3 \\ 0 & -4 & 2 & -2 & 0 \\ 0 & -6 & -2 & 3 & -7 \\ -1 & -3 & 0 & -7 & 3 \end{bmatrix}$$

Figure 2: Weighted Graph G (5, 7) (Order $|V| = 5$ and Size $|E| = 7$, Labelled Arbitrarily) and Its Associated Weighted Adjacency Matrix $WA(G)$, Degree Matrix $D(G)$ and Weighted Laplacian Matrix $WL(G)$, Respectively ($WL(G) = D(G) - WA(G)$).

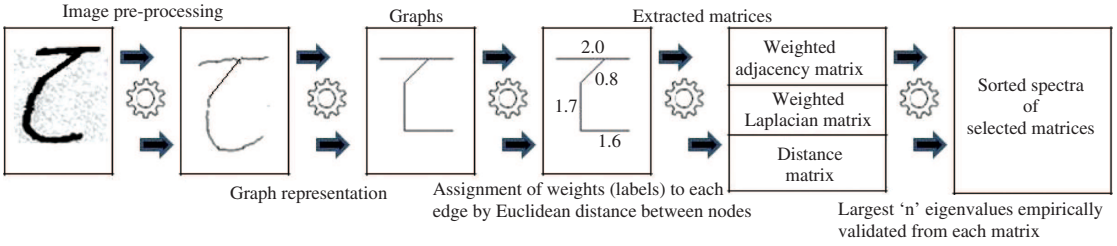


Figure 3: Process of Extraction of Sorted Spectra.

normalisation is applied to handle variability in size, and later numeral images are binarised. Finally, numeral images are skeletonised by a 3×3 thinning operator [30].

4.2 Graph Representation

There exist various graph representations [18]; however, we selected interest point graph representation as it preserves inherent structural characteristics of numeral images. It identifies the points in an image where the signal information is rich such as junction points, start and end points, and corner points of circular primitive parts of numerals. Various approaches are proposed for giving interest point graph representations. In this paper, interest point graph representation was inspired by [28, 52]. However, in contrast with [28], the edges

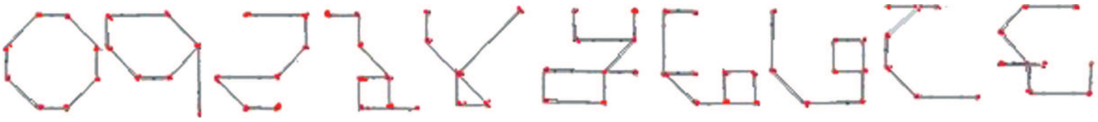


Figure 4: Snapshot of Underlying Graphs Obtained from Handwritten Devanagari Numerals with Interest Points (Numerals 0–9).

in the representation are added based on [52]. Additionally, the orientation point is further added. Figure 4 shows some extracted sample numeral graphs and interest points in each numeral graph.

4.3 Feature Extraction

Weighted graphs include more discriminating information than unweighted such as stretching of the graph [18]. In order to give weights to numeral graphs, edges are labelled with the most well-known and intuitive weighting function w: $E(G) \rightarrow R^{+}$, which assigns Euclidean distance to each edge in G. Euclidean distance is computed from respective 2D coordinates of nodes incident with each edge e in E (shown in Figure 5A). The motivation behind using such a weighting function is twofold; first, it is computationally simple and, secondly, the distance between any two objects (in this study, nodes) remains unaffected with the inclusion of more objects (nodes) in the analysis [24]. However, there is an arsenal of weighting functions described in the literature [27]; one can use any one of them. As stated earlier, SGE is described in terms of matrices associated with graphs. Selection and extraction of matrices which preserve the underlining structure or topology of the numeral graphs are indispensable. In consideration to this fact, we selected the weighted adjacency matrix ($WA(G)$), weighted Laplacian matrix ($WL(G)$) and distance matrix ($Dist(G)$). These matrices exhibit different topological information (global or local) of graphs which can be crucial for the characterisation of numeral graphs. The adjacency matrix consists of a length of edges, and it is unique for each graph (up to permutation rows and columns) that leads to isomorphism, invariance of graphs. A total number of connected components and spanning trees for a given graph is given by the Laplacian matrix. A number of spanning trees $t(G)$, in a connected graph, is a well-known invariant and leads to many more discriminating properties of the graph. The distance matrix gives the mutual pairwise distance between each node; the matrix thus formed is different for graphs having equal order [1, 17, 53, 57]. Matrix decomposition follows the subsequent representation of these matrices in terms of eigenvalues (with their multiplicities) called spectral decomposition or eigendecomposition of graphs. Let M be some matrix representation of graph G ($WA(G)$, $WL(G)$ and $Dist(G)$); then the spectral decomposition (or eigendecomposition) is $M = \Phi\Lambda\Phi^{T}$ where $\Lambda = diag(\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{|V|})$ is the ordered eigenvalues of a diagonal matrix and $\Phi = (\Phi_1, \Phi_2, \Phi_3, \ldots, \Phi_{|V|})$ is the ordered eigenvectors as columns in a matrix M. Then the spectrum (eigendecomposition) of M is the set of eigenvalues $\{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{|V|}\}$. For the eigenvalues $\{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{|V|}\}$ and the corresponding eigenvectors $(\Phi_1, \Phi_2, \Phi_3, \ldots, \Phi_{|V|})$ Equation (1) holds. The advantage of using a spectrum in characterising a graph is that eigendecomposition of various matrices associated with graphs can be quickly computed (computation of a spectrum from a matrix requires $O(n^3)$ operations, where 'n' is the order of the graph). Furthermore, the spectral parameters of a graph illustrate/specify various discriminating properties, which otherwise are exponentially computed (chromatic number, sub-graph isomorphism, perturbation of graph, number of paths of length 'K' between two nodes, number of connected components in a graph, etc.). Thus, exploiting the spectrum for the graph characterisation is clearly beneficial.

$$M\Phi = \lambda\Phi \tag{1}$$

For an illustration of eigendecomposition, let $WA(G) = M$ be the matrix representation of a graph G described in Section 3.

Equation (1) can also be written as

$$M\Phi - \lambda I\Phi = 0 \quad \Rightarrow \quad (M - \lambda I)\Phi = 0 \quad \Rightarrow \quad \det(M - \lambda I) = 0 \tag{2}$$

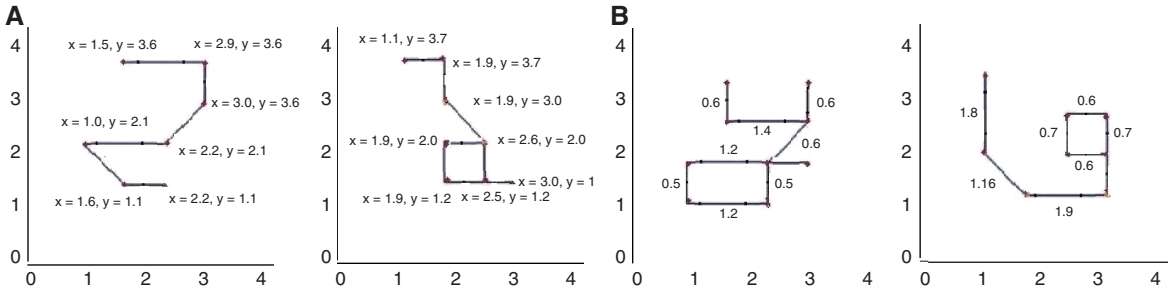


Figure 5: Illustration of Assigning Weights to Numeral Graphs: (A) Each Node Labelled with 2D Coordinates; (B) Each Edge in the Numeral Graph Labelled (Weighted) with the Euclidean Distance Between Two Adjacent Nodes.

where 'I' is the identity matrix, Φ is a special vector (eigenvector) that is in the same direction as $M\Phi$. After multiplying Φ with M, the vector $M\Phi$ is a number λ times the actual Φ, called an eigenvalue of M. That means, upon linear transformation M on Φ, λ is an amount of how much vector Φ is elongated or shrunk, reversed or unchanged, which is described by an eigenvalue.

Eigendecomposition of the weighted adjacency matrix $WA(G)$ can be carried out as follows:

$$WA(G) = \begin{bmatrix} 0 & 5 & 0 & 0 & 1 \\ 5 & 0 & 4 & 6 & 3 \\ 0 & 4 & 0 & 2 & 0 \\ 0 & 6 & 2 & 0 & 7 \\ 1 & 3 & 0 & 7 & 0 \end{bmatrix} \text{ after applying } (2) \begin{bmatrix} -\lambda & 5 & 0 & 0 & 1 \\ 5 & -\lambda & 4 & 6 & 3 \\ 0 & 4 & -\lambda & 2 & 0 \\ 0 & 6 & 2 & -\lambda & 7 \\ 1 & 3 & 0 & 7 & -\lambda \end{bmatrix}$$

Then solving the equation $-\lambda + 140\lambda^3 + 378\lambda^2 - 1445\lambda + 344$, we arrive at ordered (dominant) eigenvalues:

$$\Lambda = (12.6880, 1.9669, 0.2570, -6.0595, -8.8.523)$$

Similarly, eigendecomposition is carried out for the weighted Laplacian matrix $WL(G)$ and distance matrix $Dist(G)$. Thereafter, we arrive at feature matrices consisting of ordered (dominant) eigenvalues (spectrum) of $WA(G)$, $WL(G)$, $Dist(G)$, respectively. Furthermore, these features (spectrum) are first inspected individually for characterisation potential, and later they are fused together at decision level (or classifier level fusion) to characterise the numeral graphs.

4.4 Adequacy of the Features

Spectrum inherits different properties (global and local) from their respective graph-associated matrices which make them ideal candidates for recognition purposes; a thorough study can be found in [14, 19, 20, 21]. However, few important properties which are concerned with this study are described as follows:

- Spectrum is real if the associated graph matrix is real and symmetric. Since, the spectral decomposition map graphs in a coordinate system, any classification or clustering procedures can be used.
- Spectrum is invariant with respect to labelling of a graph (isomorphic graphs) if sorted either in ascending or descending order because swapping of two columns has no effect on values. Therefore, different orders of the graphs have no influence.
- Since each eigenvalue contains information about all nodes in a graph so it is possible to use only a certain subset of them. Therefore, it is not mandatory to use all eigenvalues. Imbalanced (short) spectra can be balanced with padding zero values.
- For disconnected graph G spectrum is the union of the spectra of different components in G.

5 Experimentation

5.1 Dataset Description and Experimental Setup

For experimentation, we used an isolated handwritten Devanagari numeral dataset from CVPR Unit, ISI Kolkata. It consists of 22,556 samples written by 1049 persons. A total of 368 mail pieces, 274 job application forms, and specially designed forms were used. In a dataset, numerals are with different writing styles, size and stroke widths. The dataset also comprises certain samples that cannot be recognised by humans also. We divided the entire dataset of labelled numeral images into three disjoint sets, namely, training, validation and test sets. The validation set is used to tune/optimise the meta-parameters of the classifier and proposed method. However, the original dataset is divided into training and testing ratios, but the authors of the dataset have stated in [10] that depending upon the requirement, the dataset can be partitioned into training, validation and test sets. Hence, we divided the dataset into two standard ratios of 60:20:20 and 50:25:25 [38] of training, validation and test sets. Figure 1 shows some numeral samples of the dataset. The complete description of the dataset can be found in [9].

Due to its robustness, which is validated from numerous fields of pattern recognition, we employed multi-class support vector machines (SVM) in association with a kernel called Gaussian kernel (also called the radial basis function, RBF-kernel) [25, 34]. There are two possible ways of classification in multi-class SVM: one-vs.-one classifier (IV1) and one-vs.-all classifier (IVA). We have used the one-vs.-one method, as it is insensitive towards an imbalanced dataset. In this method, training is done with all pairs of two-class SVMs (e.g. for 3-class problem, $1-3$, $2-3$, $1-2$), also called pairwise decomposition. All possible pairwise classifiers ($n(n-1)/2$) are evaluated and decision for unseen observation is made by majority vote. During training RBF-based SVMs have to optimise two meta-parameters (namely C and Υ, representing classification cost and non-linear function, respectively), empirically on the dataset. To arrive at optimised parameters, values for C and Υ are varied from 0.001 to 10,000 on a logarithmic scale (base-2) (i.e. 0.001, 0.01, ...). Each SVM is trained for every possible pair (C, Υ) on the training set and the recognition accuracy is tested on the validation set. Values leading to the best recognition accuracy are then used with an independent test set (Table 1).

Each spectrum (spectra of $WA(G)$, $WL(G)$ and $Dist(G)$) is investigated individually for recognition potential. From now on, we refer to the spectra of $WA(G)$, $WL(G)$ and $Dist(G)$ as (feature type) FT_1, FT_2 and FT_3, respectively.

The individual recognition results from each feature type are then compared. In order to improve the accuracy of individual classifiers, multi-classifier system (MCS) [34] or classifier fusion is employed. Classifier fusion combines their results by using various combining strategies; however, we used Bayesian fusion (described in Subsection 5.2). It is worth underlining that in MCS, individual classifiers should be accurate and diverse [34]. As stated earlier, the accuracy of SVMs is experimentally validated in a number of practical

Table 1: Class-wise Performance of All Feature Types.

Class index	Training:validation:testing						Class index	Training:validation:testing					
	60:20:20			50:25:25				60:20:20			50:25:25		
	FT_1	FT_2	FT_3	FT_1	FT_2	FT_3		FT_1	FT_2	FT_3	FT_1	FT_2	FT_3
1	0.90	0.93	0.79	0.89	0.92	0.96	6	0.75	0.74	0.75	0.74	0.72	0.74
2	0.92	0.94	0.77	0.87	0.93	0.93	7	0.68	0.81	0.80	0.67	0.80	0.79
3	0.78	0.72	0.73	0.72	0.71	0.72	8	0.88	0.85	0.94	0.87	0.84	0.93
4	0.69	0.85	0.93	0.68	0.84	0.92	9	0.65	0.77	0.69	0.64	0.76	0.68
5	0.81	0.67	0.96	0.80	0.66	0.95	10	0.61	0.62	0.88	0.60	0.61	0.87

(FT_1 $C = 0.125$ $\Upsilon = 0.001$, FT_2 $C = 0.031$ $\Upsilon = 0.0004$, and FT_3 $C = 0.001$ $\Upsilon = 0.004$). FT_1 = feature type one or sorted spectrum of the weighted adjacency matrix, FT_2 = feature type two or sorted spectrum of weighted Laplacian matrix and FT_3 = feature type three or sorted spectrum of distance matrix, respectively. Values of C and Υ are the validated meta-parameters for RBF-kernel SVM for each feature type FT_1, FT_2 and FT_3, respectively.

recognition problems; diversity means each classifier should make different errors or their decision boundaries should be different. In this study, diversity is achieved by using different feature types (as discussed in Subsection 4.3) of the numeral graphs.

5.2 Fusion Technique

We used the Bayesian combination rule (also known as Bayesian belief integration) as a combined technique. It is based on the concept of conditional probability. To compute the conditional probabilities of each classifier for all classes, the confusion matrix has to be calculated first. Let C^l be the confusion matrix for each classifier e_l, with $l = 1, \ldots, L$, where L is the total number of classifiers used (in this study L=3).

$$C^l = \begin{bmatrix} C_{11} & C_{12} & \ldots & C_{1N} \\ C_{21} & C_{22} & \ldots & C_{2N} \\ C_{31} & C_{31} & \ldots & C_{3N} \\ \vdots & \vdots & \ddots & \vdots \\ C_{N1} & C_{N2} & & C_{NN} \end{bmatrix} \tag{3}$$

where $i, j = 1, \ldots, N$, N is the number of classes, and $C_{i,j}$ in C^l is the total number of samples in which classifier e_l predicted class label j whereas the actual label was i. By using information present in the confusion matrix, the probability that the test sample 'x' corresponds to class 'i' if the classifier e_l predicts class j can be calculated as follows:

$$P_{ij} = P(x \in i | e_l(x) = j) = \frac{{C^l}_{i,j}}{\sum_{i=1}^{N} {C^l}_{i,j}} \tag{4}$$

The probability matrix P^l for each classifier e_l is

$$P^l = \begin{bmatrix} P_{11} & P_{12} & \ldots & P_{1N} \\ P_{21} & P_{22} & \ldots & P_{2N} \\ P_{31} & P_{32} & \ldots & P_{3N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N1} & P_{N2} & \ldots & P_{NN} \end{bmatrix} \tag{5}$$

Based on P^l for each classifier a combined estimate value, $b(i)$ for each class 'i' is calculated for each sample 'x' in the test set.

$$b(i) = \frac{\prod_{l=1}^{L} P_{i,jl}}{\sum_{i=1}^{N} \prod_{l=1}^{L} P_{i,jl}} \tag{6}$$

For a test sample, 'x' classifier e_l predicts class label j_l. To make a decision, one of the class maximum values in $b(i)$ is used.

5.3 Experimental Results

Several experiments were carried out for all three feature types (FT_1, FT_2 and FT_3) and subsequently repeated for 50 random trials of training, validation and testing in the ratios of 60:20:20 and 50:25:25, respectively. In each trial, the performance of the proposed method is assessed by the recognition rate in terms of F-measure,

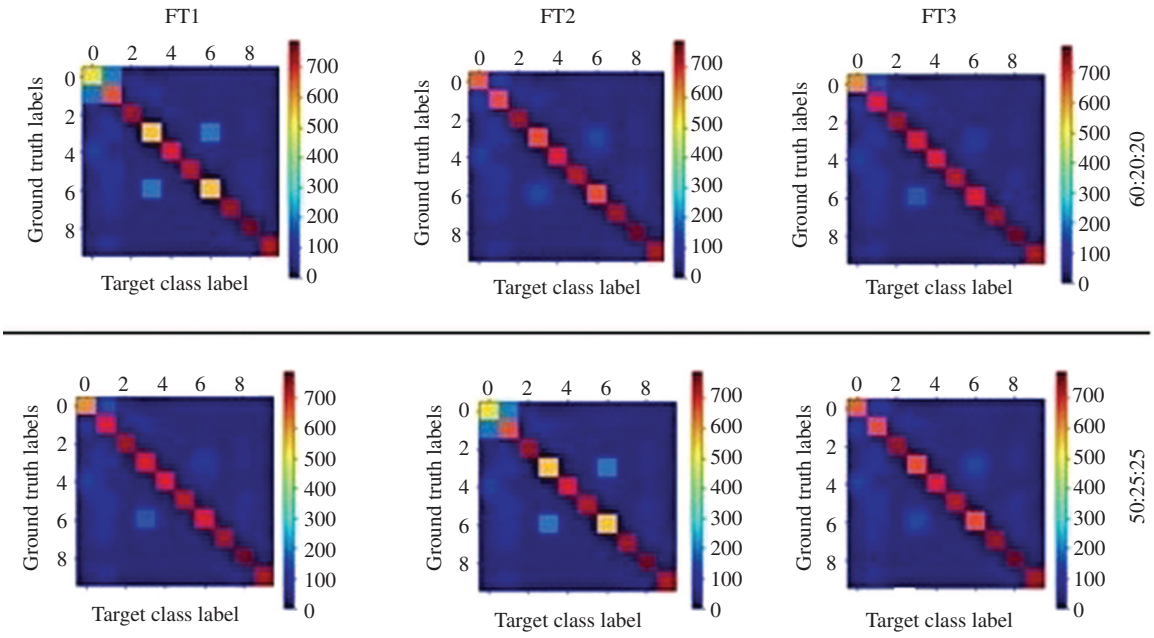


Figure 6: Confusion Matrices for Each Feature Type (FT_1, FT_2 and FT_3) for Both Divisions, respectively.

and the average F-measure is computed from all trials. Table 1 gives the class wise performance in terms of F-measure (for both the ratios belonging to all the feature types) and also presents validated meta-values for the RBF-kernel. Figure 6 shows confusion matrices obtained for optimised parameters of the classifier (for each feature type: FT_1, FT_2 and FT_3).

The performance of any recognition method is assessed in terms of precision, recall, and F-measure described as follows:

$$\text{Precision} = \frac{CP}{CP + FP'} \tag{7}$$

$$\text{Recall} = \frac{CP}{CP + FN} \tag{8}$$

$$F\text{-measure} = \frac{(2 * \text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{9}$$

Measures 'Precision', 'Recall' and 'F-measure' are based on correct positive, false negative, false positive, and correct negative for overall samples of the test set.

Table 2 presents the average F-measure computed from all trails. Individually, these feature types (FT_1, FT_2 and FT_3) generate 75–85% average recognition rate. Since FT_3 comprises all the pairwise distances, the shape of the numeral graph is not preserved. Numeral graphs with an equal number of vertices $|V|$ are only distinct in pairwise distances of the vertices but equal in a number of non-zero entries. Perhaps, this could be the reason for its (FT_3) lowest recognition result (75–76%). FT_1 and FT_2 preserve the exact shape of the numeral graphs such as the presence of edges and also their weights; hence they generate over 80% average recognition rates. Since each graph-associated matrix contains non-overlapping information, therefore by combining the classifiers at the decision level greater recognition rates can be achieved. With classifier fusion at the decision level, we achieved the maximum average recognition rate (fusion is carried out individually for each trial and then average recognition accuracy is recorded) of 93.73%, as shown in Table 3. Therefore, by decision fusion at the classifier level recognition rate is increased (FT_1, FT_2 and FT_3) by 7.9%. The numerals which have the same underlying graph structure (more or less) build the misclassified pairs such as Devanagari zero and Devanagari one (as can be observed from Table 1, confusion matrices and Figure 7). Furthermore, the invariance property of the spectrum also adds to the confusion. It can be understood by observing the shape of the Devanagari numeral three and Devanagari numeral six (as shown in Figure 7, just mirror images

Table 2: Overall Average Recognition Performance (in Terms of *F*-Measure) for Both Ratios.

Dataset	Feature type	Ratios of training, validation and testing	Overall recognition rate
CVPR Unit, ISI Kolkata	FT_1	60:20:20	85.83 ± 1.05
		50:25:25	84.63 ± 1.16
	FT_2	60:20:20	83.93 ± 0.98
		50:25:25	82.73 ± 0.86
	FT_3	60:20:20	76.73 ± 0.96
		50:25:25	75.83 ± 0.99

Table 3: Average Recognition Rate.

Dataset	Ratios of training, validation and testing	Average recognition rate in terms of F-measure
CVPR Unit, ISI Kolkata	60:20:20	93.83 ± 1.12
	50:25:25	92.73 ± 0.97

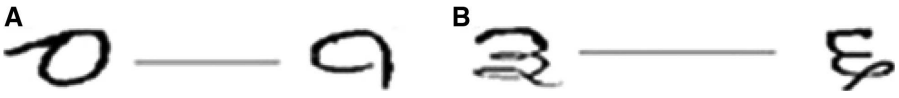


Figure 7: Few Confusing Pairs Such as (A) Devanagari Zero and Devanagari One (More or Less Same Graph Representation) and (B) Devanagari Three and Devanagari Six (Just Mirror Images of Each Other).

Table 4: Empirical Evaluation of '*n*' Largest Eigenvalues.

Ratios of training, validation and testing	Largest eigenvalues	Recognition accuracy in terms of *F*-measure
60:20:20	1	90.65 ± 0.98
	2	91.75 ± 0.95
	3	93.83 ± 1.12
	4	89.85 ± 0.93
	5	88.95 ± 0.92
50:25:25	1	89.75 ± 0.92
	2	90.85 ± 0.96
	3	92.73 ± 0.97
	4	86.75 ± 0.91
	5	85.65 ± 0.94

of each other). As, we sorted the spectrum, therefore, their spectra are more or less equal. In consideration of these facts, recognition performance is encouraging.

It should be noted that each spectrum was sorted in descending order. In order to choose 'n' largest eigenvalues for each feature type FT_1, FT_2 and FT_3, we conducted experiments for various values 'n' on the validation set. We observe that only the small value of 'n' has significant development ($n = 3$). But when we increase the value of 'n' we do not observe much significant development in recognition performance. Thus, in experimentation, we considered the value of 'n' equal to 3 for every feature type (FT_1, FT_2 and FT_3). The results obtained after fusion with varying 'n' are shown in Table 4.

5.4 Comparative Study

We compared our model with the paper, in which graph representation is used on the same dataset. From the literature, we observe that the authors in [8] achieved a recognition accuracy of 95.85% (in terms of

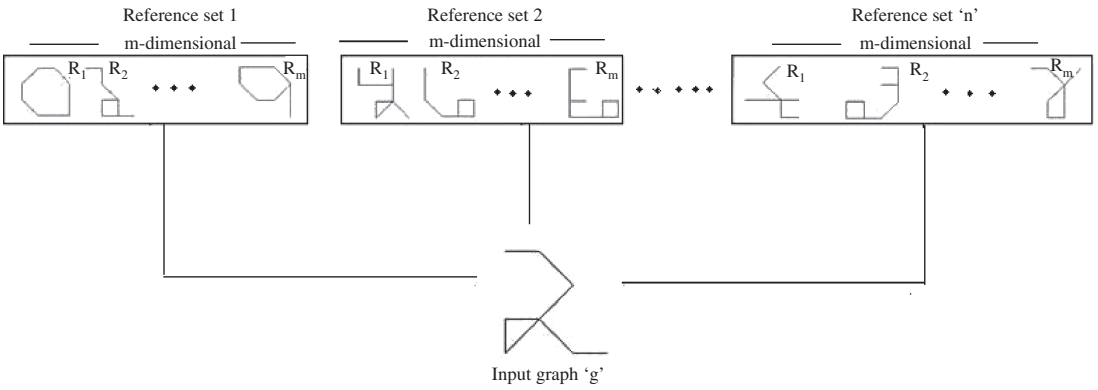


Figure 8: Illustration of the Compared Model.

F-measure) by using graph representation and Lipchitz embedding. Lipchitz embedding is based on transforming a graph into '*n*' distances to already set aside '*n*' *m*-dimensional reference sets of graphs, as shown in Figure 8. Each '*di*' in the feature vector $F = (d_1, d_2, \ldots, d_n)$ is obtained by taking the minimum distance between the input graph '*g*' and graphs present in each reference set, that is, $d_i = \min(R_1, R_2, \ldots, R_m)$, where $R_1, R_2, \ldots, R_m$ are the individual graphs belonging to each reference set. Consequently, a graph '*g*' is converted to the *n*-dimensional vector space R^n by computing the graph edit distance (GED) of '*g*' to all of the '*n*' reference sets (each *m*-dimensional). However, transforming numeral graphs into vector spaces by computation of dissimilarities from '*n*' *m*-dimensional selected reference sets (carefully selected set of graphs) is time-consuming. The input graph '*g*' is matched with every single graph in the reference set that requires time complexity in cubic order with respect to the order of the graph (thus inappropriate for graphs having large orders). Furthermore, GED depends on optimisation of various factors, namely, insertion, deletion and substitution cost of nodes and edges. Recognition performance is greatly influenced by the number and dimension of reference sets. Moreover, the type of the graphs selected from the dataset for each reference set also has a great impact on the recognition performance. Our model transforms numeral graphs into vector space by eigendecomposition (or spectrum of a numeral graph as a feature vector) to avoid computationally expensive pairwise graph matching. Besides being powerful in characterising small graphs, they are easy to compute (computational complexity is $O(n^3)$, where '*n*' is the number of nodes present in a graph) and include information about the structure (shape) of the graphs. Furthermore, most misclassification occurs in our model due to the invariance property of the spectrum. Thus the efficacy of the proposed method can easily be justified. Since our model gives graph representation, it is not directly comparable with conventional feature representation models.

6 Conclusion and Future Work

In this study, we presented a method that exploits robust graph representation and SGE for recognition of style variant, cursive handwritten characters by taking a case study of Devanagari numerals. Largest '*n*' eigenvalues (spectrum) are extracted from selected (application-dependent) weighted numeral graph-associated matrices. We empirically validated highest performing '*n*' from each spectrum. Recognition performance from individual spectra ranges from 75 to 85% (in terms of the average *F*-measure). In order to augment recognition accuracy classifier fusion at the decision level is also studied. That increases recognition accuracy significantly, as shown in Table 4. The performance of the method is corroborated by conducting extensive experiments on the standard CVPR Unit, ISI Kolkata dataset. After observing the results from different experiments, we conclude that the proposed method is effective in representing complex relationships between different primitives, different intra-class size, style, image transformations (translation, scale, rotation, reflection and mirror image) and cursiveness for recognition of handwritten Devanagari numerals. However, the

method may not withstand handwritten characters/numeral if they have the same (more or less) underlying graph representation. Furthermore, the invariance property of the spectrum also adds to the confusion. Hence, due to these reasons, most misclassification occurs.

There are various issues that need further investigation. For example, there seems to be room for employing the spectra of the further graph-associated matrices at decision level fusion. Furthermore, experiments/observations in this study have been based on SVMs. It would be interesting to repeat experiments/observations with different classifiers. Moreover, using probabilistic outputs (Fuzzy) in one-vs.-one and one-vs.-all multi-class classification seems to be an interesting topic for further research. Finally, in this study, we have used Euclidean distance for labelling graphs. It would be interesting to observe the influence of distance on eigendecomposition of numeral graphs.

Acknowledgement: We would like to thank Prof. Ujjwal Bhattacharya and Prof. B.B. Chaudhuri of Computer Vision and Pattern Recognition Unit (CVPR-Unit) of Indian Statistical Institute (ISI) Kolkata for providing the Handwritten Devanagari Numeral dataset.

Bibliography

[1] An Eigendecomposition Approach to Weighted Graph Matching Problems, 1988. http://cognitrn.psych.indiana.edu/rgoldsto/papers/weighted%20graph%20match2.pdf.
[2] S. Bag and G. Harit, A survey on optical character recognition for Bangla, *Sadhana* **38** (2013), 133–168.
[3] R. Bajaj, L. Dey and S. Chaudhury, Devnagari numeral recognition by combining decision of multiple connectionist classifiers, *Sadhana* **27** (2002), 59–72.
[4] N. P. Banashree and R. Vasanta, OCR for script identification of Hindi (Devanagari) numerals using feature sub selection by means of end-point with neuro-memetic model, *International Journal of Computer, Electrical, Automation, Control and Information Engineering* **1** (2007), 206–210.
[5] N. Banashree, D. Andhre and R. Vasanta, OCR for script identification of Hindi (Devanagari) numerals using error diffusion Halftoning Algorithm with neural classifier, in: *Proceedings of World Academy of Science, Engineering and Technology*, pp. 46–50, 2007.
[6] S. Basu, N. Das, R. Sarkar, M. Kundu, M. Nasipuri and D. Kumar, A hierarchical approach to recognition of handwritten Bangla characters, *Pattern Recognit.* **42** (2009), 1467–1484.
[7] M. I. Bhat and B. Sharada, Recognition of handwritten Devanagari numerals by graph representation and SVM, in: *2016 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2016*, pp. 1930–1935, 2016.
[8] M. I. Bhat and B. Sharada, Recognition of handwritten Devanagari numerals by graph representation and Lipschitz embedding, in: K. Santosh, M. Hangarge, V. Bevilacqua and A. Negi, eds., *Recent Trends in Image Processing and Pattern Recognition. RTIP2R 2016. Communications in Computer and Information Science*, vol. 709, Springer, Singapore, 2017.
[9] U. Bhattacharya and B. B. Chaudhuri, Databases for research on recognition of handwritten characters of Indian scripts, in: *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR. 2005*, pp. 789–793, 2005.
[10] U. Bhattacharya and B. B. Chaudhuri, Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals, *IEEE Trans. Pattern Anal. Mach. Intell.* **31** (2009), 444–457.
[11] U. Bhattacharya, S. K. Parui, B. Shaw and K. Bhattacharya, Neural combination of ANN and HMM for handwritten Devanagari numeral recognition, in: *Tenth International Workshop on Frontiers in Handwriting Recognition,* Oct. 2006, La Baule (France), Suvisoft, 2006.
[12] S. Bhowmik, S. Polley, Md. Galib Roushan, S. Malakar, R. Sarkar and M. Nasipuri, A holistic word recognition technique for handwritten Bangla words, *Int. J. Appl. Pattern Recognit.* **2** (2015), 142–159.
[13] A. E. Brouwer and W. H. Haermers, *Spectra of Graphs, Universitext*, Springer, New York, 2012.
[14] R. A. Brualdi, *A Combinatorial Approach to Matrix Theory and its Applications*. https://www.crcpress.com/A-Combinatorial-Approach-to-Matrix-Theory-and-Its-Applications/Brualdi-Cvetkovic/p/book/9781420082234.
[15] M. Cheriet, M. El Yacoubi, H. Fujisawa, D. Lopresti and G. Lorette, Handwriting recognition research: twenty years of achievement and beyond, *Pattern Recognit.* **42** (2009), 3131–3135.
[16] F. R. K. Chung, Spectral graph theory, *ACM SIGACT News* **30** (1999), 14.
[17] D. Conte, P. Foggia, C. Sansone and M. Vento, Thirty years of graph matching in pattern recognition, *Int. J. Pattern Recognit. Artif. Intell.* **18** (2004), 265–298.
[18] D. Conte, P. Foggia, C. Sansone, M. Vento, A. Kandel, H. Bunke and M. Last, *Applied Graph Theory in Computer Vision and Pattern Recognition* (*Stud. Comput. Intell.*), vol. 52, Springer-Verlag, New York, Inc., Secancus, NJ, pp. 85–135, 2007.
[19] D. Cvetkovic, P. Rowlinson and S. Simic, Eigenspaces of Graphs. Print. https://www.amazon.com/Eigenspaces-Graphs-Encyclopedia-Mathematics-Applications/dp/0521573521.

[20] D. Cvetkovic, P. Rowlinson and S. Simic, Spectral Generalisations of Line Graphs. Print. https://londmathsoc.onlinelibrary.wiley.com/doi/pdf/10.1112/S0024609305224463.
[21] D. M. Cvetkovic, M. Doob, I. Gutman and A. Torgašev, *Recent Results in the Theory of Graph Spectra*, 1991. https://www.elsevier.com/books/recent-results-in-the-theory-of-graph-spectra/cvetkovic/978-0-444-70361-3.
[22] N. Das, J. M. Reddy, R. Sarkar, S. Basu, M. Kundu, M. Nasipuri and D. K. Basu, A statistical-topological feature combination for recognition of handwritten numerals, *Appl. Soft Comput. J.* **12** (2012), 2486–2495.
[23] N. Deo, *Graph Theory with Applications to Engineering & Computer Science*. http://store.doverpublications.com/0486807932.html.
[24] M. M. Deza and E. Deza, *Encyclopedia of Distances*, 2009. http://www.uco.es/users/ma1fegan/Comunes/asignaturas/vision/Encyclopedia-of-distances-2009.pdf.
[25] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*. John Wiley, New York, Sect. 654, 2000.
[26] H. Fujisawa, Forty years of research in character and document recognition – an industrial perspective, *Pattern Recognit.* **41** (2008), 2435–2446.
[27] J. A. Gallian, A dynamic survey of graph labeling, *Electron. J. Comb.* (2009), 1–219. http://www.combinatorics.org/ojs/index.php/eljc/article/viewFile/DS6/pdf.
[28] S. Ghosh, N. Das, M. Kundu and M. Nasipuri, Handwritten Oriya digit recognition using maximum common sub-graph based similarity measures, in: S. Satapathy, J. Mandal, S. Udgata and V. Bhateja, eds., *Information Systems Design and Intelligent Applications. Advances in Intelligent Systems and Computing*, vol. 435, Springer, New Delhi, 2009.
[29] S. Ghosh, N. Das, T. Gonçalves, P. Quaresma and M. Kundu, The journey of graph kernels through two decades. *Comput. Sci. Rev.* **27** (2018), 88–111.
[30] Z. Guo and R. W. Hall, Parallel thinning with two-subiteration algorithms, *Commun. ACM* **32** (1989), 359–373.
[31] M. Hanmandlu and O. V. R. Murthy, Fuzzy model based recognition of handwritten numerals, *Pattern Recognit.* **40** (2007), 1840–1854.
[32] R. Jayadevan, S. R. Kolhe, P. M. Patil and U. Pal, Offline recognition of Devanagari script: a survey, *IEEE Trans. Syst. Man Cybern C* **41** (2011), 782–796.
[33] H. B. Kekre, S. D. Thepade, S. P. Sanas and S. Shinde, Devnagari Handwritten Character Recognition using LBG vector quantization with gradient masks, in: 2013 *Int. Conf. Adv. Technol. Eng. ICATE 2013*, pp. 1–4, 2013.
[34] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience, Hoboken, New Jersey, 2005.
[35] Y. Le Cun, Y. Bengio, Word-level training of a handwritten word recognizer based on convolutional neural networks, in: *Proc. 12th IAPR Int. Conf. Pattern Recognit. (Cat. No.94CH3440-5)*, vol. 2, pp. 88–92, 1994.
[36] L. Malik, A graph based approach for handwritten Devanagari word recognition, in: *Int. Conf. Emerg. Trends Eng. Technol. ICETET*, pp. 309–313, 2012.
[37] V. N. More and P. P. Rege, Devanagari handwritten numeral identification based on Zernike moments, in: *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, 2008.
[38] O. Nelles, Nonlinear system identification: from classical approaches to neural networks and fuzzy models, 2001. https://www.springer.com/in/book/9783540673699?token=gbgen&wt_mc=GoogleBooks.GoogleBooks.3.EN.
[39] U. Pal and B. B. Chaudhuri, Indian script character recognition: a survey, *Pattern Recognit.* **37** (2004), 1887–1899.
[40] U. Pal, T. Wakabayashi, N. Sharma and F. Kimura, Handwirtten numeral recognition of six popular Indian scripts, in: *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR.2*, pp. 749–753, 2007.
[41] P. M. Patil and T. R. Sontakke, Rotation, scale and translation invariant handwritten Devanagari numeral character recognition using general fuzzy neural network, *Pattern Recognit.* **40** (2007), 2110–2117.
[42] T. Pavlidis, Decomposition of polygons into simpler components: feature generation for syntactic pattern recognition, *IEEE Trans. Comput.* **C-24** (1975), 636–650.
[43] R. J. Ramteke and S. C. Mehrotra, Feature extraction based on moment invariants for handwriting recognition, in: *Proc.–IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1–6, Bangkok, 2006. doi: 10.1109/ICCIS.2006.252262.
[44] S. Sarkar and K. L. Boyer, Quantitative measures of change based on feature organization: eigenvalues and eigenvectors, *Comput. Vision Image Understanding* **71** (1998), 110–136.
[45] R. Sarkhel, A. K. Saha and N. Das, An enhanced harmony search method for Bangla handwritten character recognition using region sampling, in: *2015 IEEE 2nd Int. Conf. Recent Trends Inf. Syst.*, pp. 325–330, 2015.
[46] R. Sarkhel, N. Das, A. K. Saha and M. Nasipuri, A multi-objective approach towards cost effective isolated handwritten Bangla character and digit recognition, *Pattern Recognit.* **58** (2016), 172–189.
[47] M. Schmidt, G. Palm and F. Schwenker, Spectral graph features for the classification of graphs and graph sequences, *Comput. Stat.* **29** (2014), 65–80.
[48] B. Schölkopf and A. J. Smola, *Learning with kernels: Support Vector Machines, Regularisation, Optimisation, and Beyond. Adaptive Computation and Machine Learning*. The MIT Press Cambridge, Massachusetts London, England, 2002.
[49] N. Sharma, U. Pal, F. Kimura and S. Pal, Recognition of off-line handwritten Devanagari characters using quadratic classifier, in: P. Kalra and S. Peleg, eds., *Proceedings of PICVGIP 2006*, LNCS 4338, Springer-Verlag, Berlin Heidelberg, Germany, pp. 805–816, 2006.
[50] J. Shi and J. Malik, Normalized cuts and image segmentation, in: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 731–737, 1997; *IEEE Trans. Pattern Anal. Machine Intell.* **28** (2000), 888–905.
[51] P. Singh and A. Verma, Handwritten Devanagari digit recognition using fusion of global and local features, *Int. J. Multimed. Ubiquitous Eng.* **89** (2014), 6–12.

[52] M. Stauffer, A. Fischer and K. Riesen, A novel graph database for handwritten word images, Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 10029 LNCS, November, pp. 553–563, 2016.
[53] J. Stewman and K. Bowyer, Learning graph matching, in: *Comput. Vision 1988. ICCV 1988. IEEE 2nd Int. Conf.* Vol. 31, pp. 494–500, 1988.
[54] O. Trier and A. K. Jain, Torfinn, Feature extraction methods for character recognition – a survey, *Pattern Recognit.* **29** (1996), 641–662.
[55] C. Vasantha Lakshmi, R. Jain and C. Patvardhan, Handwritten Devanagari numerals recognition with higher accuracy, in: *Proc. – Int. Conf. Comput. Intell. Multimed. Appl. ICCIMA 2007*, vol. 3, pp. 255–259, 2008.
[56] P. Wang, Historical handwriting representation model dedicated to word spotting application. Computer vision and Pattern Recognition [cs.CV]. Universitie Jean Monnet – Saint-Etienne, 2014. English. NNT: 2014STET4019.
[57] R. C. Wilson, Graph Theory and Spectral Methods for Pattern Recognition. https://www.cs.york.ac.uk/cvpr/talks/PRGraphsFinal.pdf.