Neural Comput & Applic (2017) 28 (Suppl 1):S51–S63
DOI 10.1007/s00521-016-2309-5

ORIGINAL ARTICLE

Comparison of HMM- and SVM-based stroke classifiers for Gurmukhi script

Karun Verma[1] · **Rajendra Kumar Sharma**[1]

Received: 6 October 2015 / Accepted: 30 March 2016 / Published online: 20 April 2016

Abstract With the evolution of touch-based devices, development of handwriting recognition systems has received attention from many researchers. An online handwriting recognition system for Gurmukhi script is proposed in this paper. In this work, 74 stroke classes have been identified and implemented for character recognition of Gurmukhi script. Seventy-two different combinations of SVM- and HMM-based stroke classifiers with five different features have been experimented. The results of recognition of 35 basic characters of Gurmukhi script on a data set of 1750 Gurmukhi characters written by 10 writers have been reported using three best classifiers and a voting-based classifier built with the help of these classifiers. A character recognition rate of 96.7 % has been achieved using the voting-based classifier, whereas a recognition rate of 96.4 % has been achieved with an HMM-based classifier.

Keywords Gurmukhi script · SVM · HMM · Features extraction · Online handwritten character recognition

1 Introduction

Handwriting is the art of drawing letters, words, or symbols on a surface with pen or pencil. A digital handwriting system captures a handwritten character by sequence of strokes. Online handwriting recognition is the process of analyzing these strokes and determining a particular character from a selected character set with a requisite degree of confidence. With the invention of affordable pen/stylus-based devices, research in the area of online handwriting recognition has received attention from many researchers [1, 2]. A limited amount of work in online character recognition of Gurmukhi has been carried out in recent past [3, 4]. However, a lot of works in handwriting recognition have been done for simplified Chinese, traditional Chinese, English, Japanese, and Korean languages [5–7]. In the literature, isolated character recognition for many Indic languages like Bangla, Hindi, Tamil, and Telugu has been reported by many researchers [8–11]. In this paper, recognition of Gurmukhi script, a popular script in northern part of India to write Punjabi language, has been attempted. This script consists of 9 vowels (laga matras), 41 consonants, *bindī*, *ṭippī* as two symbols for nasal sounds and *addak* as a symbol to duplicate the sound of a consonant. It is written from left to right. Table 1 illustrates various characters of Gurmukhi script. A sequence of points captured or drawn during a pen-down and pen-up event, while writing, is called a stroke. Gurmukhi characters are written in various styles that magnifies the difficulty in recognizing a particular character. The strokes for writing characters in Gurmukhi script can be drawn in one of the three horizontal zones, namely, upper, middle and lower zones as depicted in Fig. 1. Most of the character symbols of Gurmukhi are written in middle zone. The region above the headline denotes the upper zone, where some of the vowels (ੰ, ੱ, ੴ) and sub-parts of some other vowels (ਿ, ੀ) reside, while the middle zone represents the area below the headline where the consonants and some sub-parts of vowels (ਿ, ੀ) are present. The lower zone represents the area below middle zone where some vowels and certain

✉ Karun Verma
karun.verma@thapar.edu

Rajendra Kumar Sharma
rksharma@thapar.edu

[1] Computer Science and Engineering Department, Thapar University, Patiala 147004, Punjab, India

Table 1 Chart of Gurmukhi characters

ੳ	ਅ	ੲ	ਸ	ਹ	9 Matras	ਾ
ਕ	ਖ	ਗ	ਘ	ਙ		ਿ
ਚ	ਛ	ਜ	ਝ	ਞ		ੀ
ਟ	ਠ	ਡ	ਢ	ਣ		ੁ
ਤ	ਥ	ਦ	ਧ	ਨ		ੂ
ਪ	ਫ	ਬ	ਭ	ਮ		ੇ
ਯ	ਰ	ਲ	ਵ	ੜ		ੈ
ਖ਼	ਗ਼	ਜ਼	ਫ਼	ਲ਼		ੋ
ਵ਼	41 Consonants					ੌ
◌ੰ	◌ਂ	◌ੱ	3 Nasal Symbols			

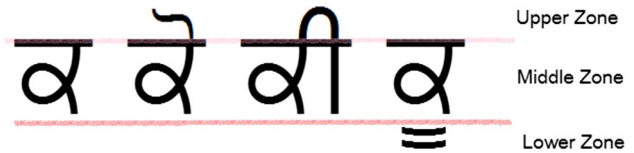


Fig. 1 Zones in Gurmukhi script

half characters lie in the foot of consonants. A Gurmukhi character can be written with a single or a combination of strokes, with stroke considered as the smallest unit. A major problem in character recognition of Gurmukhi is detection of headline and baseline. Appearance of similar looking strokes in these three zones results in formation of different characters which escalates the issue. This paper focuses on creation of a single classifier for estimation of strokes in the three zones. The solution has further been augmented with a robust postprocessing mechanism to transform the set of strokes into Gurmukhi characters.

This paper is organized into seven sections. This section introduces basic concepts behind the work. Section 2 describes work done in the area of handwriting recognition for various languages. Section 3 describes features of Gurmukhi script and how handwritten data for Gurmukhi script are collected. This section also discusses various stages of a Gurmukhi character recognition system. Section 4 explains how various features have been calculated and then organized for recognition of strokes after applying various preprocessing steps on the collected data. Section 5 describes two classifiers, namely support vector machine (SVM) and hidden Markov model (HMM) that have been used for classification of strokes in this work. This section also explains how the models are constructed from the training sets of various features as obtained in Sect. 4. Section 6 describes the postprocessing steps performed to generate Gurmukhi characters from the identified strokes. Section 7 concludes the work done with salient findings.

2 Related work

In the computing environment, keyboards had been an effective interface for interacting with computers, but with advancements in computational technology, and advent of touch-based systems, a more natural way of interacting with the computers is being explored. Handwriting is one such natural way that was pioneered and patented by Yaeger et al. [12]. This section briefly reviews the literature on handwriting recognition systems. Plamondon and Srihari [13] in their survey on online and offline handwriting recognition systems explained various processes involved, starting from input to final understanding/recognition of characters for a language. They have discussed categories of recognition methods, namely structural/rule-based methods and statistical methods in their work in order to recognize handwriting with pen based devices. Structural/rule-based methods involve with defining robust and reliable rules for recognition purposes. In Statistical methods, shape of a graphical mark known as stroke is described by fixed number of features, and the classes of these graphical marks are described by multidimensional probability distribution. Various recognition methods have been experimented by researchers across the world for handwriting recognition of scripts. These methods include artificial neural network (ANN), hidden Markov model (HMM), support vector machine (SVM), dynamic time warping (DTW), elastic matching and rule-based methods. Table 2 illustrates the recognition methods used and accuracy achieved by various researchers for different languages.

3 Data collection and steps in recognition process

Verma and Sharma [4] identified and grouped the strokes as per the zone in which they appear. The number of stoke classes considered by them was 102. They took 13 stroke classes in upper zone, 82 stroke classes in middle zone and 7 stroke classes in lower zone, and proposed zone-wise classifiers in their work. In this study, a different approach for classification has been tried where a single classifier is built with all the stroke classes, irrespective of their zone of appearance. In this process, similar looking strokes have been merged and a common strokeID has been assigned to them irrespective of their zones. For example, 123 (ੑ) and 131 (ੵ) have been merged and assigned a common strokeID 123; 122 (ੱ) and 132 (ੱ) have been merged and assigned a common strokeID 122; 133 (ੴ) and 134 (ੴ) have been merged and assigned a common strokeID 133. This resulted in a total of 74 stroke classes, which have been used for training of classifiers in this work.

Table 2 Recognition techniques used and accuracy achieved for different languages

Language	References	Technique	Accuracy
Arabic	Almuallim and Yamaguchi [1]	Elastic matching	91.0 % Character recognition accuracy on a data set of 400 words written by two persons
Bangla	Bhattacharya and Pal [9]	Stroke segmentation and recognition from Bangla online handwritten text. SVM used for stroke recognition	97.7 % Stroke recognition accuracy
	Biswas et al. [10]	HMM-based stroke recognizer	91.9 % Character recognition accuracy
	Bhattacharya et al. [14]	Multilayer perceptron classifier	83.6 % Stroke recognition accuracy
	Bhowmik et al. [15]	SVM-based hierarchical classification scheme	85.1 % Character recognition accuracy
	Dutta and Chaudhury [16]	Feed forward neural network	90.0 % Accuracy for numeral characters and 85.0 % Accuracy for alphanumeric characters
	Parui et al. [17]	HMMs	84.6 % Stroke recognition accuracy
English	Garcia et al. [18]	Neural networks, extended to HMM	82.0 % Symbol recognition accuracy
	Hu et al. [19]	HMM-based stroke recognizer	94.5 % Word recognition accuracy
Hindi	Bharath and Madhavanath [8]	Lexicon free and Lexicon-driven recognizer	87.1 % Character recognition accuracy with combined recognizers at 20K lexicon size
	Connell et al. [20]	HMM- and nearest neighbor-based stroke recognizer	86.5 % Character recognition accuracy
	Joshi et al. [21]	Subspace method for classification	94.5 % Character recognition accuracy
	Bhattacharya et al. [22]	ANN- and HMM-based classifiers	91.3 % Stroke recognition accuracy
Japanese	Jager et al. [5]	HMMs	Word recognition rate of 81.0 % Without language modeling and 86.0 % with bigram modeling
	Takahashi et al. [23]	HMMs	90.0 % Character recognition accuracy
Kannada	Prasad et al. [24]	Rule-based classifier, divide and conquer	81.0 % Character recognition accuracy
Punjabi	Lehal and Singh [25]	Offline character recognition using nearest neighbor technique	96.0 % Accuracy
	Kumar and Sharma [3]	Postprocessing algorithm, and stroke sequencing for character formation, stroke recognition using SVMs	95.6 % Character recognition accuracy
	Sharma et al. [26–28]	HMMs and elastic matching technique	91.9 % Character recognition accuracy
	Kumar et al. [29]	Postprocessing algorithm, and stroke sequencing for character formation, stroke recognition using SVMs	95.0 % Character recognition accuracy
	Verma and Sharma [4]	Zone-based features for stroke recognition	92.1 % Stroke recognition accuracy
Tamil	Bharath and Madhavanath [30]	Lexicon-driven recognizer	91.8 % Character recognition accuracy
	Joshi et al. [31]	DTW	94.8 % Character recognition accuracy
	Aparna et al. [32]	Finite-state automaton for stroke recognition	82.0 % Character recognition accuracy
	Sundaram and Ramakrishanan [11]	Attention feedback System for stroke segmentation. SVMs used for stroke classification	98.1 % Symbol recognition accuracy
Telugu	Jayaraman et al. [33]	Baseline detection, SVM-based classifiers for different sets of strokes divided by baselines	83.0 % Character recognition accuracy

3.1 Data collection

In this work, we have addressed the problem of identification of characters of Gurmukhi script. A character in Gurmukhi script can be written with one or more stroke combinations. One of the major steps in recognition of Gurmukhi characters is recognition of these strokes. Based on recognized strokes, and their combination with nearby strokes, the final Gurmukhi character is formed. To start with, a stroke classifier needs to be trained for recognition

Table 3 Gurmukhi strokes used for classification

Stroke									
StrokeID	106	121	122	123	124	126	128	133	141
Sample count	86	125	138	137	90	92	97	124	124
Stroke									
StrokeID	144	145	146	148	149	150	151	152	153
Sample count	138	131	128	131	124	134	130	132	136
Stroke									
StrokeID	154	155	156	158	159	160	161	162	164
Sample count	115	118	107	126	141	133	94	130	107
Stroke									
StrokeID	165	166	167	168	169	170	171	172	173
Sample count	120	130	130	122	138	108	124	128	116
Stroke									
StrokeID	174	175	176	177	179	180	181	182	183
Sample count	127	125	135	134	123	135	134	129	131
Stroke									
StrokeID	184	185	187	188	190	191	193	194	195
Sample count	125	108	138	132	123	115	106	121	136
Stroke									
StrokeID	196	197	198	200	201	202	203	205	207
Sample count	135	102	121	135	133	108	135	129	121
Stroke									
StrokeID	208	209	210	212	215	216	222	223	224
Sample count	130	135	125	125	130	136	135	134	100
Stroke									
StrokeID	225	226							
Sample count	100	99							

of strokes. For this purpose, a total of 44301 samples of Gurmukhi words have been collected from 124 writers using Tablet PC device.

For each word, all strokes along with their writing sequence have been recorded in an XML format. For each stroke, x–y traces of digital PEN on the touch screen between successive pen-down and pen-up events have been recorded. The writers selected here were well-versed in writing Gurmukhi script. In order to have variations in handwriting samples, the writers were selected from

various regions of Punjab (INDIA). For the purpose of training of classifiers, the strokes that were written by writers as per their correct script formation in an unconstrained environment were selected and annotated. A total of 74 stroke classes, as mentioned above, have been identified. Table 3 contains these strokes, the strokeIDs associated with them and the number of samples of these strokes used for training.

The developed online Gurmukhi character recognition system has been tested on the data collected from 10 new users. These users wrote each Gurmukhi consonant five times, giving a set of 1750 Gurmukhi characters. This data set is referred as *testdata* in Sect. 6.

3.2 Steps involved in recognition of characters

This section includes a brief description of the steps involved in recognition of Gurmukhi characters.

(a) **Stroke capturing**: As the first step, we capture the *x*–*y* traces of each stroke written by the writers.

(b) **Preprocessing**: After capturing a stroke, the *x*–*y* traces are preprocessed for extraction of features in the next step. During preprocessing, a noise or distortion which arises due to software limitations is removed from original stroke. Normalization and centering of the stroke, where the input stroke is fitted into a fixed size window and is moved to the center location, is then performed. During writing of a stroke, some points may be missed by the touch sensor. These missing points are interpolated using Bézier interpolation technique. Jitters are removed from the stroke by averaging each point with its neighbors. After performing the preprocessing steps, we obtain 64 preprocessed *x*–*y* traces of the stroke in three different-sized windows, i.e., 200×200, 300×300, and 400×400, to investigate the effect of windows size on the performance of stroke recognition.

(c) **Feature extraction**: Features required for classification of a stroke as per model chosen are extracted from 64 preprocessed *x*–*y* traces, which are used for recognition of stroke. The features used in this work are elucidated in Sect. 4.

(d) **Classification**: SVM- and HMM-based classifiers have been used for classification of strokes. The results for recognition of strokes using radial basis function kernel in SVM and HMM for different feature sets are presented in Sect. 5.

(e) **Postprocessing**: Zone identification is other important step in character formation, as certain strokes based on their appearance in different zones produce different characters. Zone of a stroke is identified based on its relative position compared to other strokes. Certain other postprocessing steps like rearrangement of strokes, stroke grouping/merging are applied on these recognized strokes to form the characters.

4 Feature extraction

Five different features, namely normalized *x*–*y* traces (N_{xy}), region-based features (R_{xy}), curvature features (C_{xy}), curvature feature-based classes (C_{xy}^{N}), and directional features (D_{xy}), have been considered for classification models. This section now explains briefly how these features have been extracted from the preprocessed *x*-*y* traces.

– **Normalized *x*–*y* traces** (N_{xy}): The 64 preprocessed traces are considered as first feature set. This feature set contains 128 elements for a given stroke sample and is referred as N_{xy} feature set.

– **Region-based features** (R_{xy}): As explained in Sect. 3.2, each stroke is normalized to windows of size 200×200, 300×300 and 400×400. The windows were further divided into 100 small equi-sized subwindows and labeled as $w_1, w_2, \ldots, w_{100}$. Each point in 64 preprocessed traces will lie in one of these 100 subwindows. The window number where the point lies is taken as a feature value. As such, this feature set will have 64 elements and is referred as R_{xy} feature set.

– **Curvature features** (C_{xy}): Curvature is defined as the degree to which something is curved. The 64 normalized *x*–*y* traces have been used to calculate curvature at each point. The curvature at a point (x, y) for the curve $y = f(x)$ is given by Eq. (1) [34].

$$R_{\text{curve}} = \frac{\left(1 + \left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)^2\right)^{\frac{3}{2}}}{\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}} \tag{1}$$

The curvature at a point is calculated using three adjacent points. Since we need at least three points to calculate second-order derivative $\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}$, we shall get a feature vector of size 62 from preprocessed *x*–*y* traces. The features thus obtained are referred as C_{xy}.

– **Curvature feature-based classes** (C_{xy}^{N}): The values of curvature, C_{xy}, lie in the range $(0, \infty)$. These values have further been divided into 20 discrete classes based on uniform frequency of curvature values, and these classes have been assigned a code. Table 4 illustrates the coding pattern of these discrete classes along with

Table 4 Coding of curvature features in C_{xy}^{N}

Range of curvature	Class code	Range of curvature	Class code
(0–7.497]	1	(162.750–212.801]	11
(7.497–15.257]	2	(212.801–289.494]	12
(15.257–23.442]	3	(289.494–420.068]	13
(23.442–33.532]	4	(420.068–646.050]	14
(33.532–44.610]	5	(646.050–1103.272]	15
(44.610–58.231]	6	(1103.272–2201.842]	16
(58.231–75.771]	7	(2201.842–9049.293]	17
(75.771–96.462]	8	(9049.293–2072473.000]	18
(96.462–124.900]	9	(2072473.000–202294064.000]	19
(124.900–162.750]	10	(202294064.000–∞)	20

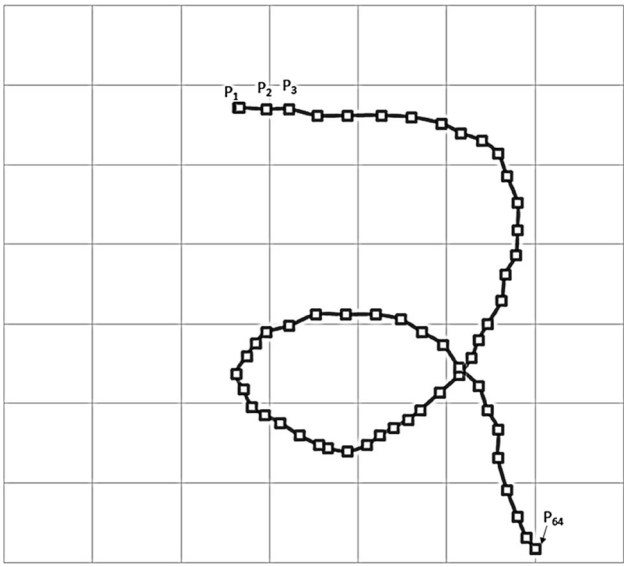
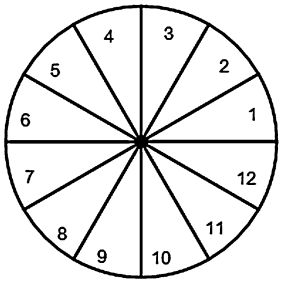


Fig. 2 Sequence of 64 preprocessed x–y traces

range of curvature values. This feature set will contain 62 such values representing the class in which the curvature value appears and is referred as C_{xy}^{N}.

– **Directional features** (D_{xy}): To calculate directional features, 64 preprocessed x–y traces ($P_1, P_2, \ldots, P_{64}$) on the trajectory of a stroke as shown in Fig. 2 have been used. The angle between two points P_i and P_{i+2}, $i = 1, 2, \ldots, 62$ is calculated and coded using a quantized vector consisting of 12 different values as illustrated in Table 5. This feature is referred as D_{xy}.

Table 5 Coding of directional features

Code	Angle range	Coding chart
1	$0° < \theta \leq 30°$	
2	$30° < \theta \leq 60°$	
3	$60° < \theta \leq 90°$	
4	$90° < \theta \leq 120°$	
5	$120° < \theta \leq 150°$	
6	$150° < \theta \leq 180°$	
7	$180° < \theta \leq 210°$	
8	$210° < \theta \leq 240°$	
9	$240° < \theta \leq 270°$	
10	$270° < \theta \leq 300°$	
11	$300° < \theta \leq 330°$	
12	$33°0 < \theta \leq 360°$	

5 Classification models

The features extracted from a stroke have now been used to build a classifier for recognition of the stroke. In this section, we explain two different classifiers experimented for classification of strokes. There are 9129 samples that have been used to train the recognition engine. The two classification approaches are given in following subsections.

5.1 Support vector machine (SVM)

For the classification process, SVM with radial basis function (RBF) kernel has been used. Other parameters of SVM that have empirically been optimized in this work are learning rate (γ) and penalty parameter (C). Three preprocessed data sets having window size 200×200, 300×300, and 400×400 have been used to obtain features sets. Five different features, explained in Sect. 4, have been obtained from these preprocessed data sets. These 15 data sets were experimented for classification using k-fold cross-validation for three different values of k as mentioned in Table 6. Table 6 also contains the values of other parameters used for training SVM-based classifiers. As such, 45 different combinations have been tested with these parameters.

Table 6 SVM parameters

Parameter	Options/values considered
Number of folds (k)	3, 4, 5
Kernel	RBF
Learning rate ($\gamma = 2^m$)	m varies from 3 (−2) −15
Penalty parameter ($C = 2^l$)	l varies from −5 (2) 15
Tolerance limit of termination (ϵ)	0.001

It is well established that scaling of inputs affects the performance of SVM. Experiments have also been conducted to find an efficient range for scaling the input parameters that yields better accuracy. These experiments suggest to fix the scale of interval as [1, 9] since highest accuracy was achieved for this scaling. The results obtained using five features with selected SVM parameters are illustrated in Table 7. Table 7 also gives the accuracy of the model on data consisting of 920 stroke samples. These 920 samples have not been used for training of SVM classifiers.

5.2 Hidden Markov model (HMM)

The features, namely R_{xy}, C_{xy}^N, and D_{xy} as explained in Sect. 4, have been used to build HMM model $\lambda = (\pi, A, B)$ for each stroke as proposed by Rabiner and Juang [35]. The other two features, N_{xy}, and C_{xy}, have not been used for building HMM models due to their large observation space. In N_{xy} feature set, the number of observations equals the dimension of window size used (i.e., 200 in case of 200×200), whereas in the case of C_{xy} feature set, number of observations cannot be fixed as the feature is a floating point value. The parameters π, A, and B for each strokeID have been calculated for the feature sets. For R_{xy} feature set, the observation set is $O = \{1, 2, \ldots, 100\}$ and the set of states is $S = \{1, 2, \ldots, 64\}$. For training HMM model for C_{xy}^N features, the observation set is $O = \{1, 2, \ldots, 20\}$, and the set of states is $S = \{1, 2, \ldots, 62\}$. For building HMM model with D_{xy}, the observation set is $O = \{1, 2, \ldots, 12\}$, and the set of states is $S = \{1, 2, \ldots, 62\}$. Algorithm 1 has been used to create HMM models from the three features sets. Algorithm 2 has now been used to test a sample for its matching strokeID.

Table 7 Feature-wise accuracy using SVM- and HMM-based classifiers

Scheme	Feature	Accuracy (%) with SVM				Accuracy (%) with HMM			
		k-fold cross-validation			Unseen data (920 samples)	k-fold cross-validation			Unseen data (920 samples)
		3	4	5		3	4	5	
200×200	N_{xy}	94.9	95.1	95.2	95.1	–	–	–	–
	C_{xy}	33.0	33.0	33.0	33.0	–	–	–	–
	D_{xy}	83.9	84.0	84.1	84.0	88.6	89.0	90.6	89.8
	C_{xy}^N	38.3	39.5	39.7	34.4	94.8	95.1	95.5	94.6
	R_{xy}	88.6	89.3	89.4	89.1	71.1	81.9	93.3	92.3
300×300	N_{xy}	95.3	96.1	97.5	96.5	–	–	–	–
	C_{xy}	37.3	37.9	38.4	35.6	–	–	–	–
	D_{xy}	86.4	87.3	88.6	85.3	89.4	91.3	92.6	90.4
	C_{xy}^N	38.0	38.3	38.5	36.9	94.4	95.1	96.3	95.0
	R_{xy}	79.0	79.3	79.7	76.9	70.3	82.5	95.6	95.5
400×400	N_{xy}	92.3	92.4	92.4	92.4	–	–	–	–
	C_{xy}	33.6	33.6	33.7	33.6	–	–	–	–
	D_{xy}	82.3	82.3	82.4	82.3	87.3	88.4	89.9	89.4
	C_{xy}^N	37.8	37.9	38.1	37.4	93.3	94.3	95.4	94.9
	R_{xy}	67.1	67.7	68.2	66.3	74.4	82.4	93.5	93.0

Algorithm 1: Algorithm for finding parameters A, B, π of HMM $\lambda = (A, B, \pi)$

S is the set of states $\{1, 2, ..., N\}$ based on feature set of size N.
O is the set of observations $\{1, 2, ..., M\}$.
π is initial probability matrix of dimension $1 \times M$, where π_{1i} = expected number of times observation i appears in state 1 for a stroke $\forall\ i \in O$.
A is transition probability matrix. This is defined as $[a_{ij}]_{M \times M}$, where a_{ij} = (number of times j^{th} observation succeeds i^{th} observation) / (number of times i is observed) $\forall\ i, j \in O$.
$B = [b_{pq}]_{M \times N}$ is state observation matrix, where b_{pq} = (number of times p is observed at state q) / (number of times observation p appears for all strokes at state q) $\forall\ p \in O, q \in S$.

Algorithm 2: Algorithm for Recognition of Stroke from HMM $\lambda = (A, B, \pi)$

N is the total number of stroke classes in HMM.
M is the total number of states in HMM.
π_i is initial probability matrix for stroke class i.
A_i is transition probability matrix for stroke class i.
B_i is the state observation matrix for each stroke class i.
$P_{max} = 0$, $i = 1$
repeat
  $P = \pi_i$, $j = 1$
  repeat
    $P = P \times A_j \times B_j$
    if $P > P_{max}$ **then**
      $P_{max} = P$
      $RS = i$
    end if
  until $j \leq M$
until $i \leq N$

As such 27 different combinations have been tested. The results obtained using the HMM models with three features are presented in Table 7. Table 7 also shows the accuracy of these models on unseen data of 920 stroke samples, which were not used for training the models.

5.3 Summary of stroke classification results

In this work, two classification models, namely SVM and HMM, have been implemented. Five different features have been experimented with these classification models, resulting into eight feature–classifier combinations. Each of these combinations has been tested for three different preprocessed window sizes, i.e., 200×200, 300×300, and 400×400. As shown in Table 7, feature–classifier combinations in 300×300 window size are performing better than feature–classifier combinations in other two window sizes. In six of the eight feature–classifier combinations, the models resulted in a performance of more than 82.3 %. The features C_{xy} and C_{xy}^{N} have a low performance ($<$39.7 %) for SVM-based classifier. The N_{xy} feature yields the highest stroke recognition accuracy of 96.5 % among SVM-based classifiers. In HMM-based classifiers, features R_{xy} and C_{xy}^{N} achieved stroke recognition rates of 95.5 and 95.0 %, respectively. Direction feature D_{xy} yields

Table 8 Some issues in character formation caused by zoning

Combination	Stroke, StrokeID (Zone: U=Upper; M=Middle; L=Lower)			Character formed
	1	2	3	
1	੫ , 159 (M)			ਪ
2	੫ , 159 (M)	‾ , 121 (M) / 126 (M)		ਖ
3	੭ , 160 (M)			ਖ
4	੭ , 160 (M)	‾ , 121 (U)		ਥ
5	੫ , 159 (M)	‾ , 121 (M) /126 (M)	‾ , 121 (U)	ਥ
6	੫ , 159 (M)	‾ , 121 (U)		ਧ
7	੫ , 159 (M)	◡ , 121 (L) 126 (L)		ਪੁ
8	੫ , 159 (M)	◡ , 126 (U)		ਪੱ
9	੍ ,224 (M)	੍ ,225 (M)		ਲ
10	੍ , 224 (U)	੍ , 225(M)	) , 197(M)	ਨੋ
11	੮ , 149 (M)	੍ , 225(M)		ਏ
12	੮ , 149 (M)	੍ , 225(M)		ਣ
13	੨ , 161 (M)	। , 162(M)		ਪ
14	੨ , 161 (M)	। , 162(M)		ਟਾ
15	੭ , 152 (M)	‾ , 121(U)		ਸ
16	੭ , 152 (M)	‾ , 121(U)		ਮ

the stroke recognition accuracy of 85.3 and 90.4 %, respectively, when used with SVM and HMM. It has also been observed that a very large training set is required to train a HMM model having features with large observation set, to minimize sparsity in transition and observational probability matrices of HMM. The two features N_{xy} and C_{xy} could not be used in training owing to data set used in this work. The three best feature–classifier combinations obtained from these experiments are C^N_{xy}-HMM, R_{xy}-HMM, and N_{xy}-SVM with stroke recognition accuracies of 95.0,

Table 9 Character-wise recognition accuracies of different models

Gurmukhi Character	Recognition accuracy of model (in %)				Confusing Characters
	$N_{xy} - SVM$	$R_{xy} - HMM$	$C_{xy}^{N} - HMM$	Voting based	
ੳ	100.0	100.0	100.0	100.0	
ਅ	100.0	100.0	100.0	100.0	
ੲ	90.0	90.0	90.0	90.0	ਣ
ਸ	90.0	92.0	92.0	90.0	ਮ
ਹ	90.0	86.0	92.0	94.0	ਰ
ਕ	100.0	100.0	100.0	100.0	
ਖ	94.0	98.0	96.0	98.0	ਧ
ਗ	84.0	80.0	82.0	88.0	ਰਾ,ਹਾ
ਘ	92.0	94.0	98.0	92.0	ਪਾ
ਙ	98.0	100.0	100.0	100.0	
ਚ	98.0	100.0	100.0	100.0	
ਛ	98.0	96.0	98.0	100.0	
ਜ	100.0	100.0	100.0	100.0	
ਝ	94.0	94.0	96.0	92.0	ਕੱ
ਞ	94.0	94.0	96.0	92.0	◌ਾਵ
ਟ	100.0	100.0	100.0	100.0	
ਠ	100.0	100.0	100.0	100.0	
ਡ	100.0	100.0	100.0	100.0	
ਢ	100.0	100.0	100.0	100.0	
ਣ	90.0	90.0	90.0	88.0	ੲ
ਤ	100.0	100.0	100.0	100.0	
ਥ	90.0	82.0	94.0	86.0	ਧ,ਟਾ,ਖ
ਦ	100.0	100.0	100.0	100.0	
ਧ	92.0	92.0	86.0	98.0	ਪ,ਖ
ਨ	100.0	100.0	100.0	100.0	
ਪ	92.0	92.0	96.0	100.0	ਧ,ਟਾ
ਫ	98.0	98.0	98.0	98.0	ਟ
ਬ	96.0	96.0	96.0	96.0	ਵਾ
ਭ	96.0	96.0	96.0	96.0	ਤ
ਮ	94.0	94.0	94.0	94.0	ਸ
ਯ	100.0	100.0	100.0	100.0	
ਰ	88.0	86.0	84.0	94.0	ਹ
ਲ	100.0	100.0	100.0	100.0	
ਵ	100.0	100.0	100.0	100.0	
ੜ	100.0	100.0	100.0	100.0	
Overall recognition rate (%)	**95.9**	**95.7**	**96.4**	**96.7**	
Average recognition time (ms)	**77.3**	**56.2**	**55.6**	**78.5**	

Table 10 Comparison of results with earlier approaches

References	Stroke classes	Features used	Classification technique	Character accuracy (%)
Sharma et al. [26]	40	x–y traces	Elastic matching	87.4
Sharma et al. [28]	40	Direction codes	HMMs	91.9
Current work	74	N_{xy}, C_{xy}, D_{xy}, R_{xy}, C_{xy}^{N}	SVMs, HMMs	96.7

95.5, and 96.5 %, respectively, for window size of 300×300.

6 Postprocessing and character generation

The three best feature–classifier combinations resulted from the experiments illustrated in Sect. 5 were further used to implement the character recognition system. A voting-based classifier using these three classifiers have also been tested. After recognition of strokes using a classifier, the strokeIDs are processed further to generate characters of Gurmukhi script in postprocessing phase. As defined in Sect. 1, middle zone of Gurmukhi script is the busiest zone and hence most of the stroke classes selected for recognition in Table 3 are middle zone strokes. Some stokes with similar shape appear in different zones as illustrated in Table 8. Based on appearance of these strokes in different zones, different characters are formed. A zone detection algorithm in order to detect the position of strokes relative to middle zone stroke has been implemented. In this algorithm, original x–y traces of the strokes are used for detecting their zone. The zonal information along with strokeID is passed to the character recognition process. The scheme proposed by Kumar et al. [3] has been used to recognize the Gurmukhi script characters.

Four character recognition systems developed using three best feature–classifier combinations and a voting-based classifier have been tested on *testdata*. The results obtained for this data set are depicted in Table 9. The proposed systems could achieve an average accuracy of 83.5 % for Gurmukhi character ਗ and an average accuracy of 88.0 % for character ਰ. The reason behind such a low accuracy achieved in recognition of these characters is due to confusion of strokes having strokeID 164 (◌੍), 165 (◌੍) with strokeID 155 (◌) and 156 (◌), respectively. Due to misclassification of these strokes, the resultant character is different from the ground truth. The system achieved 88.0 % accuracy for characters ਬ. The reason behind this low recognition rate is due to the stroke combinations, as mentioned in Table 8. Misidentification of zone of stroke is another common reason for low recognition rate of certain characters, namely ਥ, ਧ, and ਝ. The system achieved an average recognition accuracy of 100.0 % for the characters ਉ, ਅ, ਕ, ਜ, ਟ, ਠ, ਡ, ਢ, ਤ, ਦ, ਨ, ਯ, ਲ, ਵ, and ੜ. The average character recognition time in milliseconds (ms) are also elucidated in Table 9. The recognition time is calculated as the time taken by character recognition system from time of submission of strokes to the final character generated on screen. It is evident that HMM-based classifiers are faster than SVM-based classifiers. It is also evident from Table 9 that voting-based classifier, as expected, performs better than other classifiers with an overall character recognition accuracy of 96.7 %.

7 Conclusion

We have implemented three models for recognition of online handwritten Gurmukhi characters. These three models are based on SVM and HMM classifiers. A voting-based classification model based on these three models has also been implemented. We have also experimented with five different features in this work. The writing styles of Gurmukhi script users give rise to different shapes of strokes using which the characters are formed. We have carried out a brief study on the issues that are prominent in dealing with formation of confusing characters.

In an earlier study on online Gurmukhi script recognition, an accuracy of 87.4 % has been achieved using elastic matching technique by Sharma et al. [27]. They experimented with HMM models using 130 samples of each Gurmukhi character and achieved a recognition rate of 91.9 %. The HMM model was built on direction code-based feature as explained in Sect. 4. The feature used by them has eight different values of direction, whereas in the current work, we have used 12 different directions. Verma and Sharma [4] in their work on zone-based features for online Gurmukhi script achieved a recognition rate of 92.1 % for normalized diagonal zone-based features. In this work, we have achieved a higher recognition accuracy of 96.7 % for Gurmukhi script character recognition using a voting-based classifier. As expected, the average recognition time taken by voting-based classifier is more than other three classifiers. These studies are summarized in Table 10.

As a further work in this direction, the low recognition accuracies of some of the confusing characters in

Gurmukhi script can be improved by adding a more robust zone detection algorithm. Different classification techniques based on multilayer perceptron and linear discriminant analysis (LDA) for different feature combinations can also be tested. One can also explore the effect of training data on recognition rates of classification models.

Acknowledgments We take this opportunity to extend our special thanks to Technology Development for Indian Languages (TDIL), DeitY, MoCIT, Government of India, for sponsoring this research work.

References

1. Almuallim H, Yamaguchi S (1987) A method of recognition of Arabic cursive handwriting. IEEE Trans Pattern Anal Mach Intell 9(5):715–722
2. Bellegarda EJ, Bellegarda JR, Nahamoo D, Nathan KS (1993) A continuous parameter hidden markov model approach to automatic handwriting recognition, 1993. EP Patent 0,550,865
3. Kumar R, Sharma RK (2013) An efficient post processing algorithm for online handwriting Gurmukhi character recognition using set theory. Int J Pattern Recognit Artif Intell 27(04):1–17
4. Verma K, Sharma R (2015) Performance analysis of zone based features for online handwritten Gurmukhi script recognition using support vector machine. In: Selvaraj H, Zydek D, Chmaj G (eds) Progress in systems engineering. Volume 330 of advances in intelligent systems and computing. Springer International Publishing, Berlin, pp 747–753
5. Jäger S, Liu C-L, Nakagawa M (2003) The state of the art in Japanese online handwriting recognition compared to techniques in western handwriting recognition. Doc Anal Recognit 6(2):75–88
6. Liu C-L, Jaeger S, Nakagawa M (2004) Online recognition of Chinese characters: the state-of-the-art. IEEE Trans Pattern Anal Mach Intell 26(2):198–213
7. Tappert CC, Suen CY, Wakahara T (1990) The state of the art in online handwriting recognition. IEEE Trans Pattern Anal Mach Intell 12(8):787–808
8. Bharath A, Madhvanath S (2012) Hmm-based lexicon-driven and lexicon-free word recognition for online handwritten Indic scripts. IEEE Trans Pattern Anal Mach Intell 34(4):670–682
9. Bhattacharya N, Pal U (2012) Stroke segmentation and recognition from Bangla online handwritten text. In: IEEE 2012 international conference on frontiers in handwriting recognition (ICFHR). pp 740–745
10. Biswas C, Bhattacharya U, Parui SK (2012) Hmm based online handwritten Bangla character recognition using dirichlet distributions. In: IEEE 2012 international conference on frontiers in handwriting recognition (ICFHR). pp 600–605
11. Sundaram S, Ramakrishnan A (2013) Attention-feedback based robust segmentation of online handwritten isolated Tamil words. ACM Trans Asian Lang Inf Process (TALIP) 12(1):4
12. Yaeger LS, Richard WFI, Pagallo GM (2009) Method and apparatus for acquiring and organizing ink information in pen-aware computer systems, 21 July 2009. US Patent 7,564,995
13. Plamondon R, Srihari SN (2000) Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84
14. Bhattacharya U, Gupta BK, Parui SK (2007) Direction code based features for recognition of online handwritten characters of Bangla. In: IEEE ninth international conference on document analysis and recognition, 2007. ICDAR 2007, vol 1. pp 58–62
15. Bhowmik TK, Ghanty P, Roy A, Parui SK (2009) Svm-based hierarchical architectures for handwritten Bangla character recognition. Int J Doc Anal Recognit 12(2):97–108
16. Dutta A, Chaudhury S (1993) Bengali alpha-numeric character recognition using curvature features. Pattern Recognit 26(12):1757–1770
17. Parui SK, Guin K, Bhattacharya U, Chaudhuri BB (2008) Online handwritten Bangla character recognition using hmm. In: IEEE 19th international conference on pattern recognition, 2008, ICPR 2008. pp 1–4
18. Garcia-Salicetti S, Doizzi B, Gallinari P, Mellouk A, Fanchon D (1995) A hidden markov model extension of a neural predictive system for online character recognition. In: IEEE proceedings of the third international conference on document analysis and recognition, 1995, vol 1. pp 50–53
19. Hu J, Brown MK, Turin W (1996) Hmm based online handwriting recognition. IEEE Trans Pattern Anal Mach Intell 18(10):1039–1045
20. Connell SD, Sinha R, Jain AK (2000) Recognition of unconstrained online Devanagari characters. In: IEEE proceedings of 15th international conference on pattern recognition, 2000, vol 2. pp 368–371
21. Joshi N, Sita G, Ramakrishnan A, Deepu V, Madhvanath S (2005) Machine recognition of online handwritten Devanagari characters. In: IEEE proceedings of eighth international conference on document analysis and recognition, 2005. pp 1156–1160
22. Bhattacharya U, Parui S, Shaw B, Bhattacharya K et al (2006) Neural combination of ann and hmm for handwritten Devanagari numeral recognition. In: Tenth international workshop on frontiers in handwriting recognition
23. Takahashi K, Yasuda H, Matsumoto T (1997) A fast hmm algorithm for on-line handwritten character recognition. In: IEEE proceedings of the fourth international conference on document analysis and recognition, 1997, vol 1. pp 369–375
24. Prasad MM, Sukumar M, Ramakrishnan A (2009) Divide and conquer technique in online handwritten Kannada character recognition. In: Proceedings of the international workshop on multilingual OCR. pp 1–7
25. Lehal G, Singh C (2000) A Gurmukhi script recognition system. In: IEEE proceedings of the15th international conference on pattern recognition, 2000, vol 2. pp 557–560
26. Sharma A, Kumar R, Sharma R (2008) Online handwritten Gurmukhi character recognition using elastic matching. In: IEEE congress on image and signal processing, 2008. CISP'08, vol 2. pp 391–396
27. Sharma A (2009) Online handwritten Gurmukhi character recognition. Ph.D. thesis, Thapar University
28. Sharma A, Kumar R, Sharma R (2010) Hmm-based online handwritten Gurmukhi character recognition. Mach Gr Vis Int J 19(4):439–449
29. Kumar R, Sharma RK, Sharma A (2015) Recognition of multi-stroke based online handwritten Gurmukhi aksharas. Proc Natl Acad Sci India Sect A Phys Sci 85(1):159–168
30. Bharath A, Madhvanath S (2007) Hidden markov models for online handwritten Tamil word recognition. In: IEEE ninth international conference on document analysis and recognition, 2007. ICDAR 2007, vol 1. pp 506–510
31. Joshi N, Sita G, Ramakrishnan A, Madhvanath S (2004) Comparison of elastic matching algorithms for online Tamil handwritten character recognition. In: IEEE ninth international workshop on frontiers in handwriting recognition, 2004. IWFHR-9 2004. pp 444–449
32. Aparna K, Subramanian V, Kasirajan M, Prakash GV, Chakravarthy V, Madhvanath S (2004) Online handwriting recognition for Tamil. In: IEEE ninth international workshop on frontiers in handwriting recognition, 2004. IWFHR-9 2004. pp 438–443

33. Jayaraman A, Chandra SC, Srinivasa CV (2007) Modular approach to recognition of strokes in Telugu script. In: IEEE ninth international conference on document analysis and recognition, 2007. ICDAR 2007, vol 1. pp 501–505
34. Kline M (2013) Calculus: an intuitive and physical approach. Courier Corporation, Chelmsford
35. Rabiner LR, Juang B-H (1986) An introduction to hidden markov models. IEEE ASSP Mag 3(1):4–16