Pertanika J. Sci. & Technol. 26 (2): 671 - 690 (2018)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

The Effect of Pre-processing and Testing Methods on Online Kannada Handwriting Recognition: Studies Using Signal Processing and Statistical Techniques

S. Ramya* and Kumara Shama

Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, Karnataka, India

ABSTRACT

Pre-processing and testing methodology plays a significant role in online handwritten character recognition. Although many researchers have proposed several pre-processing and testing methods, the effect of these techniques on the recognition and comparisons among them are ignored. In this work, experiments were conducted to analyse the effect of various pre-processing and testing methods on Kannada handwritten data. The focus of the present work is to statistically quantify the effect on recognition time and accuracy through experiments using different pre-processing methods on online handwritten data processed by the Support Vector Machine (SVM). The performance of the SVM is also compared with various other training and testing methodology. The performance of the online handwriting recognition system is affected dramatically by the various pre-processing and testing methods. Stratified tenfold cross validation showed better performance for the Kannada handwritten dataset.

Keywords: Bootstrapping, cross-validation, down sampling, normalisation, online handwriting recognition, resampling, single partition testing, smoothing

INTRODUCTION

New technology like the pen-based interface plays a significant role in human-computer interaction. The challenge with using pen-enabled interfaces is to recognise user handwriting. Handwriting Recognition (HR) is a field of pattern recognition that aims to assign objects to a set of categories (Plamondon et al., 2000). In Online Handwriting Recognition (OHR) systems, handwritten data are captured as a sequence of coordinate locations during the writing process, known as digital inking. Online handwritten data are diversified due to individual writing styles and contexts

Article history:
Received: 20 April 2017
Accepted: 12 September 2017

E-mail addresses:
ramya.lokesh@manipal.edu (S. Ramya)
shama.kumar@manipal.edu (Kumara Shama)
*Corresponding Author

ISSN: 0128-7680 © 2018 Universiti Putra Malaysia Press.

(Tappert et al., 1990). To achieve robust recognition, handwritten data are subjected to pre-processing. Pre-processing is one of the important stages in handwriting analysis; it addresses the methodology of elimination of imperfections, data reduction, normalisation and resampling (Biadsy et al., 2011). Vuori et al. (2001) applied down sampling by two to reduce processing time followed by normalisation. For smoothing, a five-tap Gaussian filter was used (Namboodiri et al., 2004; Deepu et al., 2004; Joshi et al., 2005), followed by normalisation and resampling being performed. A moving average filter (Prasad et al., 2010; Babu et al., 2007) with a span of 3 has also been used for smoothing data. Elimination of duplicate points, a necessary procedure for accurate analysis, was emphasised by Bahlmann (Bahlmann et al., 2004). To make a stroke equi-spaced along the arc length, a resampling size of 60 was used by Prashanth et al. (2007); a resampling size of 30 has also been reported (Prasad et al., 2007; Babu et al., 2007).

Many researchers have proposed various methodology for pre-processing to achieve better recognition accuracy. The widely implemented pre-processing techniques in literature are smoothing, resampling and normalisation. Down sampling and duplicate point removal are incorporated to reduce data content. Though these are common pre-processing methods, variation in the type of smoothing techniques with various window spans and resampling with different resample size are also used. It is essential to meticulously select the pre-processing methods for the given data set to enhance the performance of the classifier for character recognition. However, there is less effort into investigating the effect of these pre-processing methods in terms of character recognition accuracy and processing time. Therefore, efficacy experiments are conducted on the pre-processing techniques, namely, smoothing, normalisation, resampling, down sampling and removal of redundant points. Various combinations of pre-processing methods are tested and their effect on recognition accuracy and processing time with a single classifier for the Kannada handwritten data set is statistically quantified.

Testing and training is yet another critical phase of the recognition system that provides the recognition estimation of the classifier. In the field of online handwriting recognition, generally, experiments are conducted either with a single partition or K-fold cross validation (Namboodiri et al., 2004; Bharath et al., 2009). Studies on various testing and training methods, namely Holdout, K-fold cross validation (with or without stratification), random sub-sampling and bootstrapping have been done (Kohavi et al., 1995; Claudia et al., 2005; Kim, 2009) in various fields of research. Each training method behaves differently and this influences the bias and variance in the error. For estimating the accuracy of the classifier, low bias and low variance in error are preferable as they indicate the classification stability of the classifier. Experiments are conducted to compare these five types of methodology with recognition accuracy, bias and variance in recognition error as the parameters of comparison.

Our work focusses on the design of experiments on pre-processing and testing methodology on Kannada handwritten data as a case study and analyses their influence on the recognition system.

METHODS

Online Kannada Handwriting Recognition System

The block diagram of online Kannada handwriting recognition system is as shown in Figure 1. Kannada handwritten characters with handwriting variations in size, translation, speed and noise were collected and stored in the database. Kannada characters from the database were input to the pre-processing module. The pre-processing module did the standardisation of script to handle the variations in handwriting. Features, namely, normalised horizontal and vertical coordinates, tangent angular differences of consecutive points, writing directions and curvature were extracted from pre-processed data and used to train and test the classifier. The performance of the classifier was evaluated for the recognition accuracy and the processing time with different combinations of pre-processing and testing methodology.



Figure 1. Block diagram of online Kannada handwriting recognition system

Data Acquisition

Since the standardised online Kannada handwritten character database is not available publicly, a database was created. The script of Kannada language is syllabic in nature. It evolved from the Kadamba script (Ghosh et al., 2010). It is one of the official languages of India and the state language of Karnataka. The language has 16 vowels (Swaragalu) and 34 consonants (Vyanjanagalu) (Kunte et al., 2007), which constitute the basic Kannada character set as shown in Figure 2. To collect the Kannada character data, a graphical user interface was developed in visual C# using the Genius Mouse Pen i608X as an input device. Native users of Kannada from different age groups ranging from 10 years to 60 years of age were selected. A total of 150 data samples of each Kannada vowel and consonant was collected. Figure 2(c) shows the sample data of online Kannada handwritten characters. The work was focussed on the basic Kannada character set; hence, the dataset was restricted to 50 character classes and it can be extended to consonant-vowel modifier combinations by pre-segmentation (Prasad et.al 2009). The horizontal and vertical coordinates of pen trace with a time stamp were acquired and saved into a text file with a unique file ID. The database included characters with varying sizes and speeds of writing.

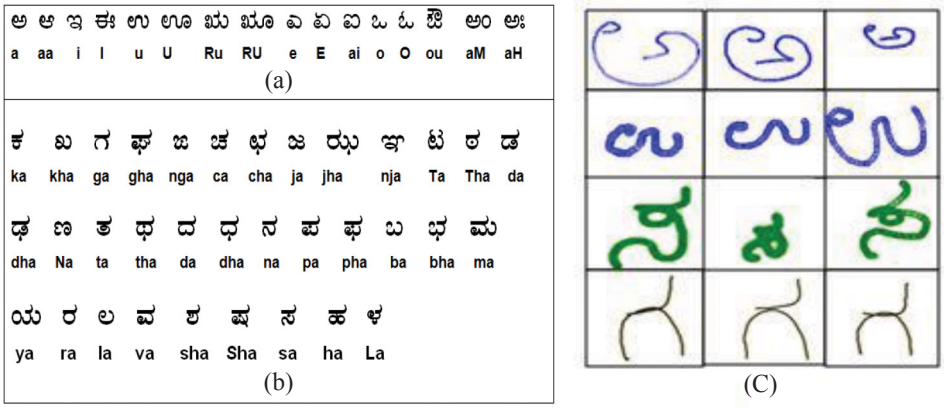


Figure 2. Kannada character set (a) Vowels (b) Consonants (c) Sample data

Pre-Processing

The raw data collected from the user are usually noisy and inconsistent. To obtain efficient classification, these data are pre-processed. In this work, the pre-processing methods, namely, normalisation, resampling, smoothing and down sampling as well as the removal of duplicated points are implemented.

Normalisation. The size and location of the input handwritten text differ from user to user. The goal of normalisation is to make data invariant to the size (Li et al., 1997). Normalised horizontal and vertical coordinates, $X_{norm(i)}$ and $Y_{norm(i)}$ are calculated using Equations 1 and 2.

$$X_{norm(i)} = \frac{(X_i - X_{min})}{(X_{max} - X_{min})} \quad [1]$$

$$Y_{norm(i)} = \frac{(Y_i - Y_{min})}{(Y_{max} - Y_{min})} \quad [2]$$

where, (X_{max}, Y_{max}) and (X_{min},Y_{min}) are the maximum and minimum horizontal and vertical coordinates of a character, allowing the data to be within the bounding box [0,1].

Resampling. The raw data captured by the digitiser are sampled uniformly in time. Resampling is performed to obtain data that are uniformly sampled along the pen trace to make the data invariant to writing speed. The original data points are replaced with regularly spaced points along the arc length using the following piecewise linear interpolation algorithm.

Algorithm: Resampling Algorithm

1. Find the Euclidean distance between the consecutive points for all the points.
2. Find the cumulative distance (D), which gives the total length of the pen trajectory.
3. Find the required spacing by dividing the distance D by the number of intervals required (Resampling size)
4. Replace the original points with a calculated set to obtain equi-spaced points.

Smoothing. While writing on the pen tablet, erratic pen movement causes shaking; this introduces noise into the signal. In smoothing, the data points of a signal are modified such that adjacent point coordinate values vary smoothly. From the literature survey, it was found that researchers generally implemented the Gaussian filter or Moving Average filter with different window spans. Gaussian smoothing computes the average of neighbouring points weighted by the Gaussian kernel. A Gaussian kernel is in the shape of a normal distribution curve (Harris et al., 1978). The input data are convolved with the Gaussian window span of N, resulting in a smoothing of data. Using Equation 3, the coefficients of a Gaussian window of size N were computed in this study.

$$W(n) = e^{-\frac{1}{2}\left(\alpha \frac{n}{N/2}\right)^2} \quad [3]$$

where, n lies $-(N - 1) / 2 \leq n < (N - 1) / 2$. The relationship between the standard deviation σ and the Gaussian probability density function is given by Equation 4.

$$\sigma = \frac{N}{2\alpha} \quad [4]$$

In a moving average filter, each data point is replaced with the mean of the adjacent data points defined within the span (Jaeger et al., 2001). The smoothening response is given by Equation 5.

$$ys_{(i)} = \frac{1}{(2M+1)}\left(y_{(i+M)} + y_{(i+M-1)} + \cdots + y_{(i-M)}\right) \quad [5]$$

where, $ys_{(i)}$ is the resultant smoothed value for the ith data point, with M as the number of neighbouring data points on either side of $ys_{(i)}$ and (2M + 1) as the filter span. In this study experiments were conducted to evaluate the performance of these filters on Kannada OHR.

Removal of duplicate points. A point is said to be a duplicate point if it has the same horizontal and vertical coordinate values as that of its adjacent point. Duplicate points result when the writing process is slower than the digitiser sampling frequency. These data are redundant points; hence, they are removed from the captured ink data. Thresholding of the horizontal and vertical coordinates' differences to zero is implemented to remove duplicate points.

Down sampling. Down sampling the data will reduce the number of horizontal and vertical coordinates in a character stroke, which will reduce the pre-processing time. Experiments were conducted in this study with down sampling by factor 2. Hence, every alternative sample was selected.

Feature Extraction and Character Recognition

Pre-processed data are the input used in the feature extraction module. The feature extraction module captures the essential characteristics of a stroke to help in recognition. Feature selection is an important task that affects the performance of the classifier (Namboodiri et al., 2004). Kannada characters are curvilinear in nature, similar to Telugu and Tamil script. Writing

direction and curvature at a point show a more promising result for Telugu and Tamil (Bharath et al., 2009; Toselli et al., 2007; Prashanth et al., 2007). Other features like normalised horizontal and vertical coordinates are also widely used in Indic script (Bharath et al., 2009). Therefore, these features were selected in this work for the online Kannada handwriting recognition.

Writing direction at a given point (x (i), y (i)) is well described using the cosine and sine angle given by the following equations.

$$\cos \propto (i) = \frac{\Delta x(i)}{\Delta s(i)} \quad [6]$$

$$\sin \propto (i) = \frac{\Delta y(i)}{\Delta s(i)} \quad [7]$$

where, Δs (i), Δx (i) and Δy (i) are defined by Equations 8, 9 and 10, respectively.

$$\Delta s(i) = \sqrt{\Delta x^2(i) + \Delta y^2(i)} \quad [8]$$

$$\Delta x(i) = x(i-1) - x(i+1) \quad [9]$$

$$\Delta y(i) = y(i-1) - y(i+1) \quad [10]$$

The curvature at a point can be calculated as:

$$\cos \beta(i) = \cos \alpha(i-1) \cos \alpha(i+1) + \sin \alpha(i-1) \sin \alpha(i+1) \quad [11]$$

$$\sin \beta(i) = \cos \alpha(i-1) \sin \alpha(i+1) - \sin \alpha(i-1) \cos \alpha(i+1) \quad [12]$$

These features are used to train the classifier. The Support Vector Machine (SVM) is widely used in object classification and recognition (Cortes et al., 1995), content-based pattern recognition, handwriting recognition, biometrics etc. SVM is based on supervised learning that analyses data and classifies the data. Many works on OHR using the SVM classifier for

different languages (Bahlmann et al., 2002) including Kannada (Rampalli et al., 2011) have been reported. Hence, the SVM classifier with RBF kernel was implemented in this study and tested for Kannada handwritten characters.

Experimental Design

A flow chart of the implementation methodology to evaluate the effect of different pre-processing methods is given in Figure 3. Numbers 1 to 4 indicate test cases. Test Case 1 focusses on performance evaluation of different resampling sizes on the recognition system. The normalised data were resampled by the following sizes.

$$\text{Resample size} = [10, 20, 30, 40, 50, 60, 70, 80, 90]$$

The normalisation and resampling methods are considered the basic pre-processing module. This module was present in all the test cases. Test Case 2 evaluated the impact of smoothing algorithms on recognition. The performance of the moving average filter and the Gaussian filter with window spans 3, 5 and 7 was tested, and their performance was compared based on recognition time and accuracy. Test Case 3 focused on down sampling the data by half, while Test Case 4 focused on analysing the effect of removing duplicate points before the basic pre-processing module.

The recognition accuracy of the classifier depends on the data used to train and test the classifier. Error estimation is a challenging task in handwriting recognition, where mainly small sample sizes with large features exist. Selecting the appropriate training and testing method is essential for evaluating the stability of the classifier. Test Case 5 compared the performance of the classifier for the following different training and testing methods.

In the single partition training-testing method, also known as 'Holdout', a pre-defined percentage of data was used to train and test the classifier. The single partition training-testing method with 70% of the data for training and 30% of the data for testing was implemented and tested. In random subsampling, the holdout procedure was repeated a number of times and the average accuracy was estimated (Kohavi et al., 1995). In this study, the holdout procedure was repeated 100 times for fair comparison (Kim et al., 2009).

In the tenfold cross-validation method, N data samples were divided into 10 groups. Then the classifier was trained with nine groups and tested with one cluster. Training and testing was repeated 10 times, with each cluster used once as a test set. Final performance estimation was done by averaging the 10 results. In stratified cross validation, the folds were divided to contain approximately equal proportions of character class.

The bootstrap is another method for accuracy estimation (Efron et al., 1993). In this method, the dataset is sampled with a replacement. When the data are drawn with a replacement, the particular sample may occur more than once in the bootstrap set. Then, the remaining data serve as the test data set.

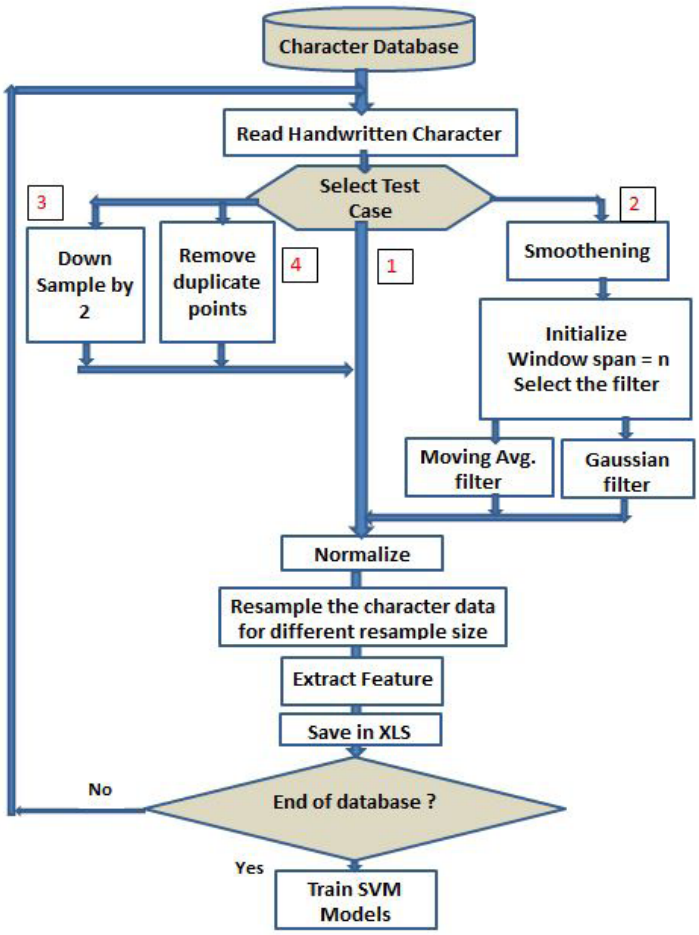


Figure 3. Flow chart of design of experiments to evaluate the effect of different pre-processing methods

Test Cases 1 to 4 were tested and evaluated using the single partition method. In Test Case 5, these results were compared with five different testing and training methods for recognition accuracy, error bias and variance to quantify the better testing-training methodology for character recognition.

RESULTS AND DISCUSSION

The effects of various pre-processing methodology were analysed and statistically evaluated. Overall recognition accuracy and processing time were parameters used for performance evaluation.

Overall Recognition Accuracy is the number of correct predictions made divided by the total number of predictions. It is defined as:

$$\text{Overall RecognitionAccuracy} = \sum_{i=0}^{n} \frac{TP_i + TN_i}{TP_i + FP_i + TN_i + FN_i} \quad [13]$$

Where,

- n is the number of samples
- True Positives (TP): Number of positive examples, labelled correctly
- False Positives (FP): Number of negative examples, labelled incorrectly
- True Negatives (TN): Number of negative examples, labelled correctly
- False Negatives (FN): Number of positive examples, labelled incorrectly

Effect of Resampling Size

To overcome variations in writing speed, the data were resampled to a fixed size, using the linear interpolation method. Resampling of data with different resampling sizes is shown in Figure 4. In some cases, like the resampling size of 10 as seen in Figure 4(b), the data get distorted, which may lead to misclassification in the recognition module. Therefore, selecting the optimum resampling size is a critical criterion for online handwriting recognition.

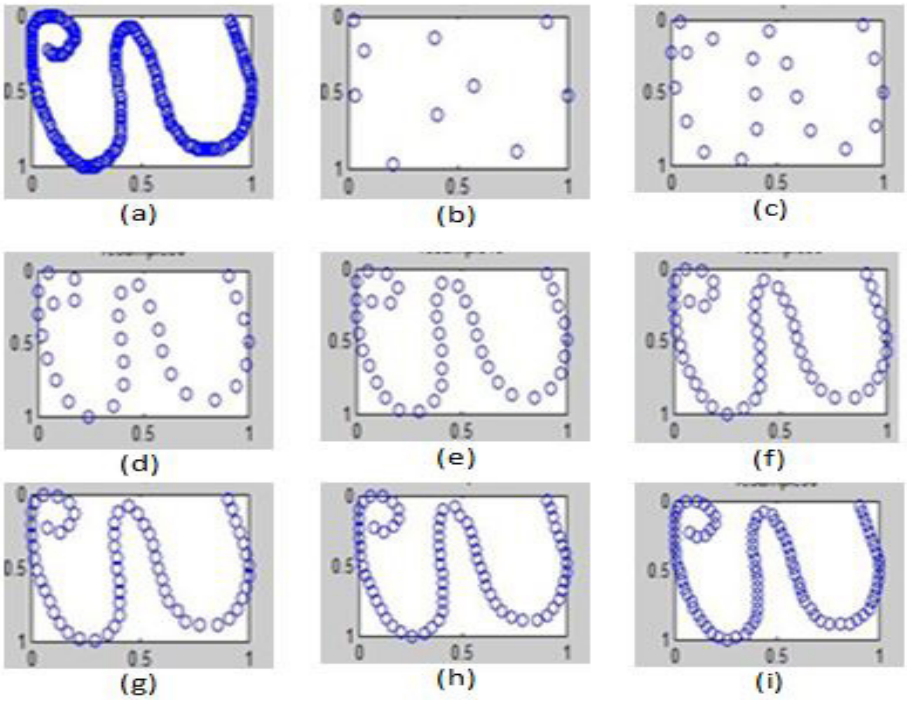


Figure 4. Effect of resampling on character shape (a) Character shape before resampling (b) – (i) Character shape for resample size 10, 20, 30, 40, 50, 60, 70, 90, respectively

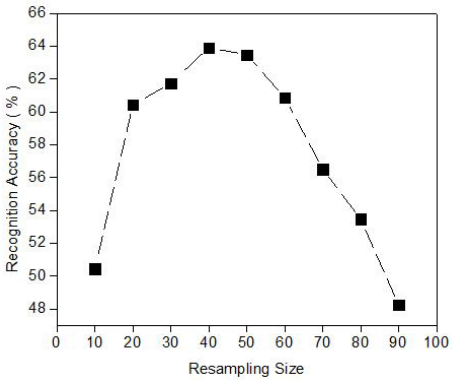


Figure 5(a). Recognition accuracy for different resampling sizes

The effect of resampling size on recognition accuracy is as depicted in Figure 5(a). When the resampling size is increased from 10 to 40, recognition accuracy improves but reduces as the resampling size reaches beyond 40. This is due to over-fitting of the features. For lower resampling sizes, recognition accuracy is low due to the distortion of the character shape, as seen in Figure 4. The resampling size of 40 exhibits the highest recognition rate of 63.91% for the Kannada dataset.

To quantify the effect of resampling size on recognition accuracy, one variable t-test was applied with the following hypothesis 'Ha' for 95% confidence.

Ha = There is no effect caused by the resampling size on recognition accuracy.

Table 1
Results of one variable t-test on resampling size

Mean	St Dev	SE Mean	T Value	P Value
57.68	5.78	1.93	29.96	0.000

Table 1 presents the results of the one variable t-test on resampling size. The mean of the data was 57.68 with a standard deviation (St Dev) of 5.78 and standard error means (SE mean) of 1.93. The t value was 29.96 with a p value of 0. Since $p<0.05$, proving that there was a significance variation in recognition accuracy with resampling size.

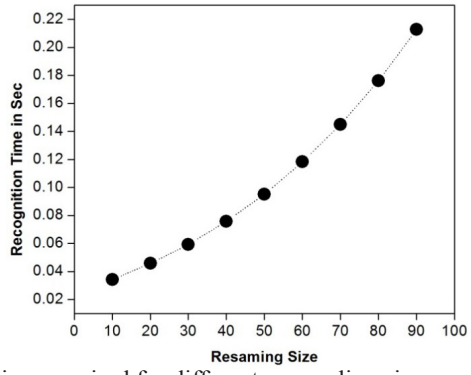
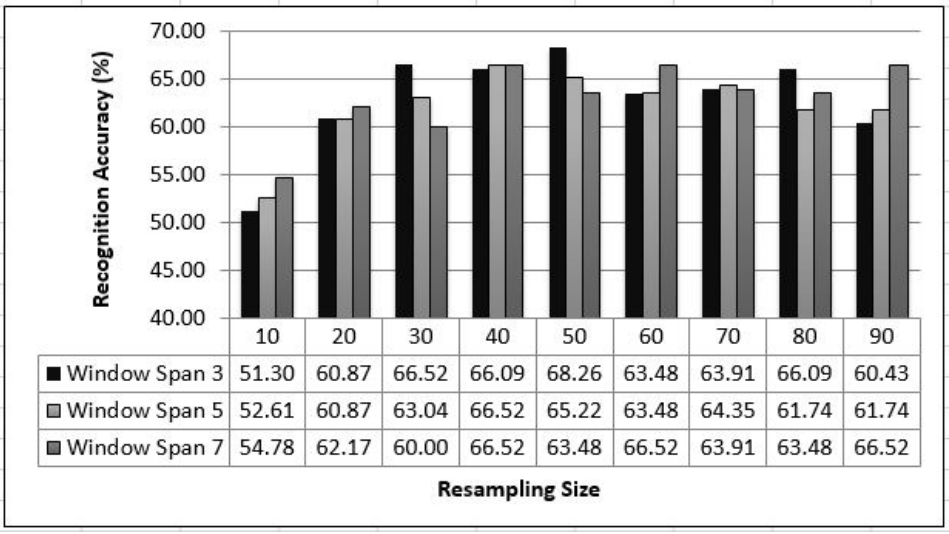


Figure 5(b). Recognition time required for different resampling sizes

Recognition time was plotted against resampling size, as shown in Figure 5(b). As the resampling size increased, it was obvious that the data samples also increased, which in turn increased the feature set length. Increase in the feature set length contributed to the increase in computation. Hence, the recognition time varied in almost linear direction with the resampling size. Although processing time increased with the resampling size, recognition accuracy did not increase with the resampling size as seen in Graph 5(a). Hence, empirical testing was required to obtain the optimum resampling size for the given dataset in general and the Kannada character set in particular. These experiments and statistical quantification signified that resampling size affected the performance of the classifier.

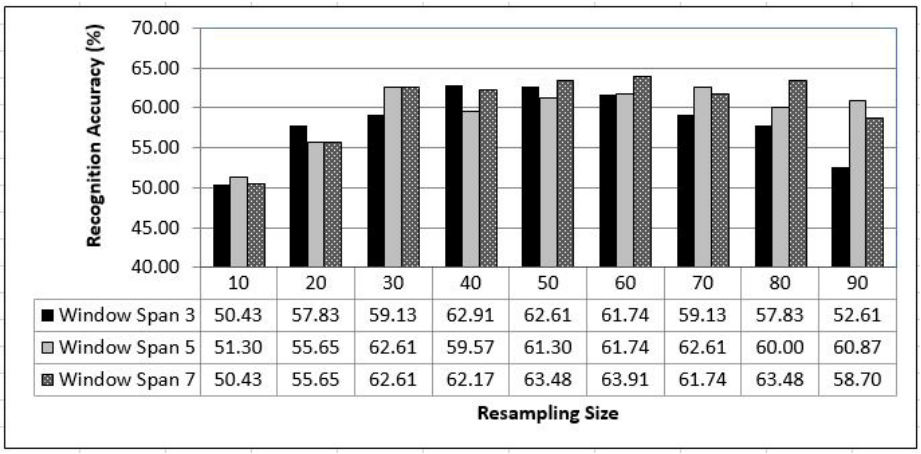
Effect of Smoothing

Due to shaking in the digitiser or due to writing style, noise can be introduced in the handwritten data. Smoothing is carried out to overcome such noise. The moving average filter and Gaussian filter of window sizes 3, 5, and 7 were implemented and tested. The recognition accuracy of the classifier after the smoothing with different filters is depicted in Graph 6(a) and 6(b). The horizontal axis represents the resampling size whereas the vertical axis represents the recognition performance. Character identification accuracy was around 50% to 70% for the different resampling sizes for both the smoothing methods.



	10	20	30	40	50	60	70	80	90
Window Span 3	51.30	60.87	66.52	66.09	68.26	63.48	63.91	66.09	60.43
Window Span 5	52.61	60.87	63.04	66.52	65.22	63.48	64.35	61.74	61.74
Window Span 7	54.78	62.17	60.00	66.52	63.48	66.52	63.91	63.48	66.52

Figure 6(a). Effect of moving average filter with different window spans on recognition accuracy



	10	20	30	40	50	60	70	80	90
Window Span 3	50.43	57.83	59.13	62.91	62.61	61.74	59.13	57.83	52.61
Window Span 5	51.30	55.65	62.61	59.57	61.30	61.74	62.61	60.00	60.87
Window Span 7	50.43	55.65	62.61	62.17	63.48	63.91	61.74	63.48	58.70

Resampling Size

Figure 6(b). Effect of Gaussian filter with different window spans on recognition accuracy

The moving average filter with window size 3 exhibited the maximum recognition rate of around 68.26% for the resampling size of 50, whereas the Gaussian filter responsed to a maximum of 63.91% for the window size 7 at the resampling size 60. The recognition rate was higher for the resampling size 40-60 range for both the smoothing filters.

Figure 7 shows the plot of recognition time required for different smoothing techniques. The processing time increases more or less in linear manner with the resampling size. The average time difference between the Gaussian filter (Gauss) with window span (WS) 7 and the moving average (mov Avg) filter with window span 3 response was 3 ms, while the Gaussian filter was faster than the moving average filter.

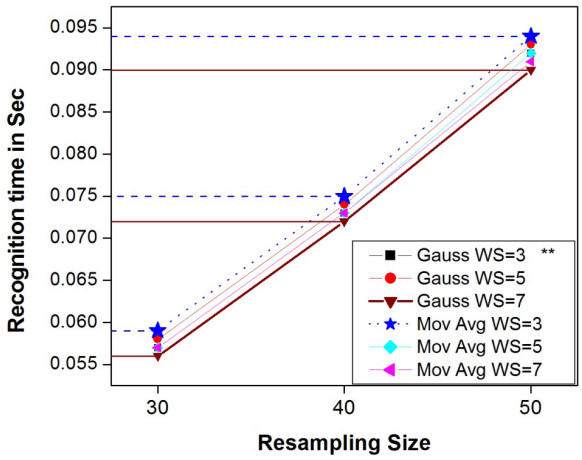


Figure 7. Recognition time variation for the different smoothing techniques
** Gauss - Gaussian filter, WS - Window Span, Mov Avg – Moving Average Filter

Effect of Removal of Duplicate Points and Down Sampling

Due to the speed variation in online handwriting, duplicate points or repetitive points were present in the data samples. Removing the duplicate points removed variation in the shape of the characters, but the length of the strokes was reduced. The number of duplicate points in our data was found to be in the range of 1 to 4 points.

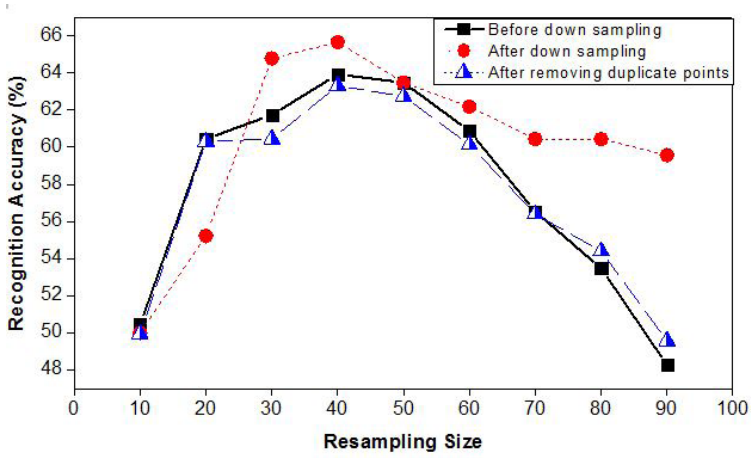


Figure 8(a). Response of recognition accuracy with and without down sampling and duplicate point removal technique

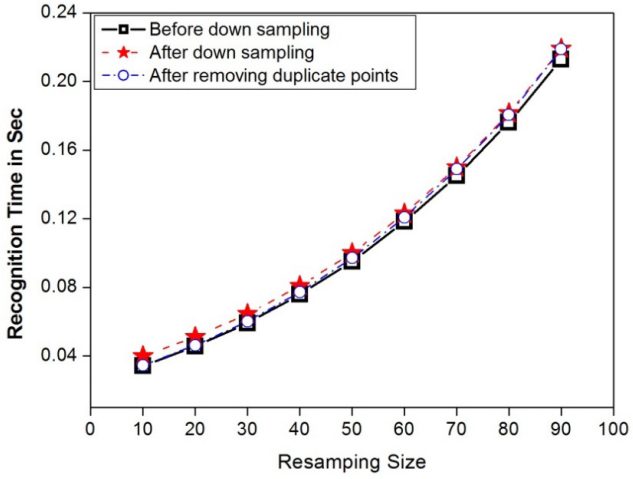


Figure 8(b). Recognition time distribution for down sampling and duplicate point removal technique

Down sampling is a method that reduces the data sample size. Down sampling by two sizes was implemented; hence, every alternative sample which reduced the signal size was selected. Figure 8(a) shows the plot of the recognition accuracy variations, before and after removing duplicate points and down sampling. From the plot, it can be seen that the performance of the

recogniser showed improvement after down sampling. However, there was little change in the recognition accuracy with the duplicate point removal method; this also happened with the processing time plot in Figure 8(b). Hence, there was no significant effect in removing duplicate points on the performance of the classifier. The processing time for down sampling was long because the data had to be down-sampled, followed by normalisation and resampling, which takes time.

Table 2
Results of paired t-test for recognition accuracy of different pre-processing methods

Pre-Processing Methods	Mean	St. Dev	SE Mean	Mean Diff	t-Value	p-Value
Removal of duplicate points	57.25	5.38	1.79	-0.435	-0.56	0.591
Down sampling	60.19	4.93	1.64	2.51	1.61	0.147
Smoothing with Gauss filter of window span 3	58.36	4.44	1.48	0.676	0.77	0.464
Smoothing with Gauss filter of window span 5	59.52	3.74	1.25	1.84	0.97	0.360
Smoothing with Gauss filter of window span 7	60.24	4.55	1.52	2.56	1.49	0.175
Smoothing with Moving Avg. filter of window span 3	63.00	5.10	1.70	5.31	3.50	0.008
Smoothing with Moving Avg. filter of window span 5	62.17	4.01	1.34	4.49	3.08	0.015
Smoothing with Moving Avg. filter of window span 7	63.04	3.80	1.27	5.36	2.66	0.029

To understand the significance of all these pre-processing methods, the paired t-test was applied. The recognition results with and without the pre-processing were compared for 95% confidence. Table 2 represents the results of various pre-processing methods. The following observation can be derived from Table 2. The highest mean of 63.04 was shown for smoothing with the moving average filter of window span 7. However, removal of duplicate points depicted the maximum deviation of 5.38. The t-value was the highest for a moving average filter with a window span of 3, indicating that the effect was significant compared to the other pre-processing methods. The minimum t-value of -0.56 was observed for the removal of duplicate points, signifying it was least significant, which was also indicated by the p value, >0.05. The p-value column indicates that the moving average filter had more significance than the Gaussian filter as the p-value was comparatively low.

Results of Different Testing and Training Methodology

Experiments with five different training-testing methods, namely, single partition random subsampling, tenfold cross-validation with and without stratification and the bootstrap methods were implemented. The classifiers tested for the moving average filter with a span of 3 and the Gaussian filter with the window size 7 because these two methods yielded the highest recognition rate in Test Case 2. The resampling size of 40, 50 and 60 were selected as these resampling sizes exhibited overall better recognition rates in all the above test cases. Table 3 lists the various pre-processing methods that were adopted to estimate the performance of the training and testing methods.

Table 3
List of different pre-processing methods used for the performance estimation of the testing and training methodology

Pre-Processing Method	Smoothing Filter	Window Span	Resampling Size
P1	Moving Average	3	40
P2	Moving Average	3	50
P3	Moving Average	3	60
P4	Gaussian	7	40
P5	Gaussian	7	50
P6	Gaussian	7	60

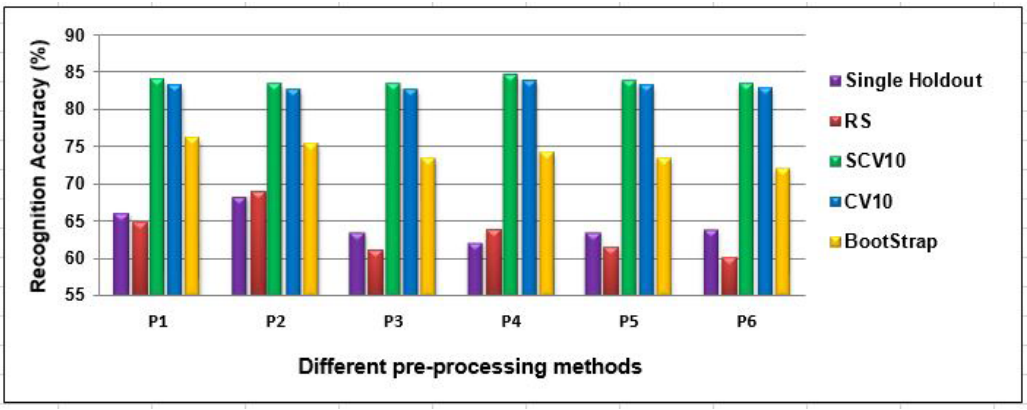


Figure 9. Comparison of recognition accuracy with the different testing and training methods

The performance of the training-testing methods is shown in Figure 9. The stratified tenfold cross validation method (SCV10) showed a better recognition result than the other estimation methods. Random subsampling (RS) showed poor recognition accuracy compared with the

other methods. Bootstrapping performance was better than the single holdout estimation. The computational time of single holdout was faster as it did involve repeated training as with the other methodology. Overall, there was a 20% increase in recognition accuracy using the stratified tenfold cross validation method compared to the single partition method for the same SVM classifier with RBF kernel. However, the recognition accuracy obtained by the training and testing method depends on the classifier used along with features drawn from the dataset.

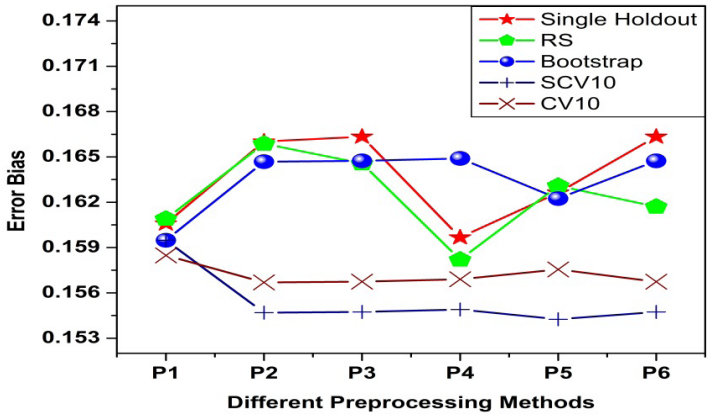


Figure 10(a). Error bias plot for the different training-testing methodologies with the pre-processing methods

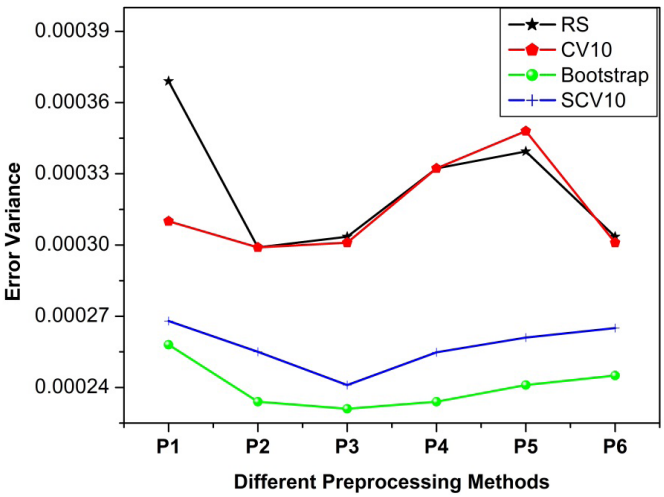


Figure 10(b). Error variance plot for the different training-testing methodologies with the pre-processing methods

The error bias and error variance plots for the different training and testing methods are shown in Graphs 10(a) and (b). The bootstrap method showed greater bias in some of the pre-processing methods. The tenfold cross validation and stratified cross validation methods showed stable bias for the different pre-processing methods compared with the other training-testing methods. Stratification reduced the variance of the cross validation as seen in the Graph 10(b). The random subsampling method had relatively high variance and bias for all the pre-processing methods. The bootstrap method had the least variance.

Our results indicated that cross validation with stratification performed better in terms of recognition, error bias and error variance compared with the tenfold cross validation method. Bootstrap estimation had low variance but showed large bias for some of the pre-processing methods. Overall, stratified cross validation performance is recommended for online Kannada handwriting recognition.

CONCLUSION

A computational analysis was carried out on the online Kannada handwriting recognition system with different pre-processing and testing methodology for a specific classifier, namely SVM, for the feature set consisting of normalised horizontal and vertical coordinates, writing direction and curvature at a point. Based on the results, the following conclusions could be drawn.

Recognition accuracy significantly depends on resampling size. The resampling size of 40 was found to be better for the Kannada character dataset used in this study. The recognition time showed more variation and was less linear with the resampling size.

Smoothing using the moving average and Gaussian filters was implemented. A moving average filter with a window span of 3 and Gaussian filter with a window span of 7 provided better recognition results. The Gaussian filter with window span 7 was 3 ms faster than the moving average filter.

Down sampling the data by a factor of 2 showed improved recognition accuracy. Removal of duplicate points had the least significance in the recognition results.

The stratified tenfold cross validation method performed better compared with the other training-testing methods. The recognition accuracy of the classifier improved from 63% to 83% when compared with the single partition method. However, the recognition accuracy obtained by the training and testing method depends on the classifier used as well as the features drawn from the dataset.

These observations will benefit researchers in choosing appropriate pre-processing and testing methodology for online handwriting recognition. Though these conclusions are drawn based on the performance of a single classifier, a comparative study of several different classifiers will provide a more generalised insight into the effect of pre-processing methods. The recognition accuracy can be improved further by selecting more features and tuning the classifiers.

ACKNOWLEDGEMENT

The authors would like to acknowledge the laboratory support rendered by Manipal Academy of Higher Education of Technology, Manipal University, Manipal, India for this research work.

REFERENCES

Babu, V. J., Prasanth, L., Sharma, R. R., & Bharath, A. (2007). HMM-based online handwriting recognition system for Telugu symbols. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition* (pp. 63–67). IEEE.

Bahlmann, C., Haasdonk, B., & Burkhardt, H. (2002). Online handwriting recognition with support vector machines – A kernel approach. In *Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition* (pp. 49–54). IEEE.

Bahlmann, C., & Burkhardt, H. (2004). The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(3), 299–310.

Bharath, A., & Madhvanath, S. (2009). Online handwriting recognition for Indic scripts. *Guide to OCR for Indic Scripts* (pp. 209–234). Springer, London.

Biadsy, F., Saabni, R., & El-Sana, J. (2011). Segmentation-Free Online Arabic Handwriting Recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, *25*(07), 1009–1033.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine learning, 20*(3), 273-297.

Deepu, V., Madhvanath, S., & Ramakrishnan, A. G. (2004). Principal component analysis for online handwritten character recognition. In *Proceedings of the 17th International Conference on Pattern Recognition* (pp. 327–330). IEEE.

Ghosh, D., Dube, T., & Shivaprasad, A. (2010). Script recognition – A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(12), 2142–2161.

Harris, F. J. (1978). On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the IEEE, 66*(1), 51-83.

Jaeger, S., Manke, S., Reichert, J., & Waibel, A. (2001). Online handwriting recognition: The NPen ++ recognizer. *International Journal on Document Analysis and Recognition*, *3*(3), 169–180.

Joshi, N., Sita, G., Ramakrishnan, A. G., Deepu, V., & Madhvanath, S. (2005). Machine recognition of online handwritten Devanagari characters. In *Proceeding of the Eighth International Conference on Document Analysis and Recognition* (pp. 1156–1160). IEEE.

Kunte, R. S., & Samuel, R. S. (2007). A simple and efficient optical character recognition system for basic symbols in printed Kannada text. *Sadhana*, *32*(5), 521-533.

Li, X., & Yeung, D. Y. (1997). Online handwritten alphanumeric character recognition using dominant points in strokes. *Pattern Recognition*, *30*(1), 31–44.

Namboodiri, A. M., & Jain, A. K. (2004). Online handwritten script recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(1), 124–130.

Plamondon, R., & Srihari, S. N. (2000). Online and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(1), 63–84.

Prasad, M. M., Sukumar, M., & Ramakrishnan, A. G. (2009, July). Divide and conquer technique in online handwritten Kannada character recognition. In *Proceedings of the International Workshop on Multilingual OCR* (pp. 11–16). ACM.

Prasad, M. M., Sukumar, M., & Ramakrishnan, A. G. (2010). Orthogonal LDA in PCA transformed subspace. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition (ICFHR)* (pp. 172–175). IEEE.

Prasanth, L., Babu, V. J., Sharma, R. R., & Rao, G. V. P., & Mandalapu, D. (2007 September). Elastic matching of online handwritten Tamil and Telugu scripts using local features. In *Ninth International Conference on Document Analysis and Recognition, 2007* (Vol. 2, pp. 1028-1032). IEEE.

Rampalli, R., & Ramakrishnan, A. G. (2011). Fusion of complementary online and offline strategies for recognition of handwritten Kannada characters. *Journal of Universal Computer Science*, *17*(1), 81–93.

Sin, B. K., & Kim, J. H. (1997). Ligature modeling for online cursive script recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(6), 623–633.

Tappert, C. C., Suen, C. Y., & Wakahara, T. (1990). The state of the art in online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(8), 787–808.

Toselli, A. H., Pastor, M., & Vidal, E. (2007, June). On-Line handwriting recognition system for Tamil handwritten characters. In *Proceedings of the 3rd Iberian Conference on Pattern Recognition and Image Analysis, Part I* (pp. 370–377). Springer, Berlin, Heidelberg.

Vuori, V., Laaksonen, J., Oja, E., & Kangas, J. (2001). Experiments with adaptation strategies for a prototype-based recognition system for isolated handwritten characters. *International Journal on Document Analysis and Recognition*, *3*(3), 150–159.