Asian Journal of Information Technology 10 (5): 180-191, 2011
ISSN: 1682-3915

Binary Validation as Segmentation for Cursive Handwriting Recognition

Hong Lee and Brijesh Verma
School of Computing Science, Central Queensland University, New South Wales and Victoria, Australia

Abstract: A novel Binary Validation as Segmentation (BVS) is presented in this study. BVS is a bottom-up approach for character segmentation in off-line cursive handwriting recognition. The character segmentation is a process to extract individual character image from a handwritten word image. The extensive literature reveals that offline handwriting recognition still suffers from the absence of reliable character segmentation algorithms. The character segmentation stage is very crucial part in offline handwriting recognition because the recognition performance is directly affected by the segmentation performance. Therefore, improving the segmentation accuracy is very high prior task in this field. BVS takes over segmentation components to generate primitives. The over segmentation guarantees that none of primitives contains image pixels belonging to different characters. Based on the primitives, neighboring primitives are combined to create evaluation hypotheses. The most competent hypothesis is sought by global competency function using neural network classifier. BVS repeats cycles of combination and evaluation iteratively. Each cycle combines one a pair of two neighboring primitives permanently. That's why the proposed approach includes the term, binary. BVS also introduces Continuous Foreground Transition (CFT) model to prevent under-segmentation errors. The proposed approach has been evaluated on CEDAR benchmark database. The results showed a significant improvement in segmentation errors. The analysis of results showed that the inclusion of CFT into the validation function has played a major role in improving over segmentation and bad-segmentation errors.

Key words: Offline cursive handwriting recognition, handwriting segmentation, neural networks, benchmark, cycle, Australia

INTRODUCTION

Offline or static cursive handwriting recognitionis an automatic process to convert an input handwritten document image into electronic character representations so, computers can manipulate the representations. Static handwriting recognition has been active research domain for decades and industrial beneficiaries have been trying to automate repetitive manpower oriented tasks such as processing postal address, bank checks, form data, historical manuscripts, etc. (Fujisawa, 2008). Despite sleepless research for decades, the performance of the state of the art systemis below the industrial standard to accommodate the real world problems. The researchers in this field agree that the main contributor of the low recognition performance is the segmentation (Suresh and Arumugam, 2007; Zhao *et al*., 2003; Arica and Yarman-Vural, 2001, 2002; Casey and Lecolinent, 1996).

Segmentation is a process to discriminate each letter from others, prior to recognition into electronic character representations. A sub-image bound by two neighboring boundaries is called a segment. Segmentation precedes the recognition. In other words, the recognition process is based on the outcomes of the segmentation process. It implies that better recognition performance can be achieved on better segmentation outcomes. However, the segmentation is very trouble some process because of the nature of handwriting which projects the informality of handwritten characters. As matter of fact, the researchers have not found any rule of thumb segmentation methods on handwritten images. That's why the segmentation process has become a notorious and chronicle contributor to the handwriting recognition field (Hussein *et al*., 1999; Sadri *et al*., 2007; Xiao and Leedham, 2000; Yanikoglu and Sandon, 1998; Elnagar and Alhajj, 2003).

Segmentation is one of the most difficult processes in handwriting so many existing approaches exclude the segmentation process and they are called holistic or segmentation free approach. Generally, holistic approach focuses on recognizing the whole word without recognizing individual characters. Holistic approach can be plausible to application domains requiring small size lexicons such as processing bank checks and postal addresses, etc. However, the main problem in holistic

Corresponding Author: Hong Lee, School of Computing Science, Central Queensland University, New South Wales and Victoria, Australia

approach is the number of classifications which is as many as the number of lexical words. In English, there are 52 alphabet characters. With segmentation approach, any words can be recognized without lexicon. On the other hand, holistic classifier needs to learn as many classes as the number of lexical words. In English dictionary, there are <200000 entries; the odds of correct classification are 1:52 versus 1:200000. Because the reason, holistic approach is not suitable for real world problems. Therefore, segmentation approach is far better approach than holistic when it comes to real world problems.

Target language: Researchers in handwriting recognition have been treating recognitions in different languages as different problems. Each language has its own character sets and the language specific syntax rules govern how those characters are to be arranged. Character segmentation requires the understanding of the target languages. The proposed BVS is only tailored to segment words in English language. Incorporating over segmentation heuristics are based on observations of English alphabet characteristics. In addition, the neural classifier for character classification is only trained on 52 English alphabet letters.

Review of segmentation techniques: Segmentation module in handwriting recognition plays a crucial role for successful performance. However, it is very difficult to find precise character boundaries without knowledge about characters. So, to avoid the error prone process, segmentation free approach has been proposed as one of the segmentation strategies. Benouareth *et al.* (2008) used Holistic method for Arabic word recognition. To build a feature vector sequence, two segmentation schemes are incorporated to divide a word into frames. The first one is uniform segmentation which vertically divides a word into equal sized frames. The second one is non-uniform segmentation. This time the size of frames varies. After word segmentation into frames, statistical and structural features are extracted by capturing ascenders, descenders, dots, concavity and stroke direction. (Vinciarelli and Luettin, 2001; Vinciarelli, 2005) used similar technique for information retrieval of writer dependent documents. Prior to the information retrieval, word recognition of the handwritten document proceeds first. Using fixed size of sliding window, density feature is extracted for HMM to perform recognition by calculating the likelihood of a word against lexicon. Similarly, methodologies presented by Gunter and Bunke (2004a, b), Su *et al.* (2009), Nopsuwanchai *et al.* (2006), Gunter and Bunke (2004a, b) and Bertolami *et al.* (2006) also adapted the segmentation free strategy. Sliding window and geometrical feature extraction are the base of the HMM module recognition.

However, the main pitfalls of segmentation free approaches heavily depend on the size of lexicon and they are only suitable for recognition domains with limited or small size lexicons. Because lexicon size greatly affect the recognition performance, reduction of unlikely lexicon words would increase the chances to find correct matching lexicon words during the recognition process. Mozaffari *et al.* (2008) proposed a lexicon reduction scheme for Static Farsi handwriting recognition by analyzing dots within characters.

Comparing to the holistic approach mentioned above, segmentation strategy is to find the character boundaries as much as possible even if it might produce unnecessary dissection points which are referred as over segmentation points. Al-Hamad and Zitar (2010) developed neural based segmentation strategy using frameworks of over segmentation and validation. Viard-Gaudin *et al.* (2005) made an effort to model the writing sequence of strokes for a handwritten image. Following the writing order, the word image is segmented into individual strokes. Then, K-means algorithm clusters the strokes into representative symbols. The recognition is to model the sequence of the symbols by incorporating HMMs. Kim *et al.* (2002) proposed a segmentation strategy by analyzing types of connectivity between neighboring digit letters. In their approach, they identified 6 types of connectivity. Prior to segmentation, it proceeds to locate multiple break points based on the connectivity type. Then, the complete segmentation is to be made by checking the combinations of segments. It is worth noting that their approach is not effective when there are >3 touching points between digits. Liu and Gader (2002) proposed an approach with over segmentation and Dynamic Programming (DP) strategy. A word image is segmented into sub-images which represents a single or partial character. Function of union between neighboring sub-images assigns the confidence value based on the compatibility. The compatibility accounts for the spatial relationships and relative sizes between neighboring unions by neural networks. Recognition by DP is to find a sequence of union that fits a given lexicon string. Verma *et al.* (2004) adapted over segmentation strategy by locating handwriting features such as upper and lower word contours, holes, upper and lower contour minima and vertical density histograms. Based on heuristics regarding handwriting features, confidence values are assigned to each over-segmented point. Two neural validators are incorporated for further validation. The first one inspects

the characteristics of each segmentation point. The second one validates each segmentation point based on the knowledge of characters.

A heuristic technique proposed by Blumenstein and Verma (2001) locates structural features and over-segments each word. For each segmentation point, a left primitive (preceding the segmentation point) and a joined primitive (joined on the segmentation point) are extracted along with other features from the segmentation region. Those features are fed to segmentation point validation neural networks to produce a number of confidence values. The final segmentationwas decided based on the confidence values. Similar technique, rule-based over segmentation and validation is used by Verma (2003). Rule-based modules validate every over segmentation point against closed area, average character size, contour code of left character and density. Verma *et al.* (2001) also proposed over segmentation and validation approach to solve the cursive handwriting recognition problem. Three different validation experts are employed to produce the ranks and confidence values which are the ingredients of Borda count. The final word recognition is the lexical word with the highest Borda count. In Xu *et al.* (2003), over segmented primitives of cursive handwritten month words to process handwritten bank cheques are fed to HMM and neural networks to find the optimum segmentation paths. Vellasques *et al.* (2008) distinguishes the validation from filtration of the over segmentation points. They also defines that the excessive amount of unnecessary over segmentation point directly affects the recognition performance. Therefore, it is emphasized that the filtration of the over segmentation points should precede the validation. In their touching digit recognition system, they show that filter can remove up to 83% of the unnecessary over segmentation hypothesis.

BINARY VALIDATION AS SEGMENTATION

The proposed Binary Validation as Segmentation (BVS) is a bottom-up approach to build up correct characters from primitives.

Architectural overview: In Fig. 1, the architectural overview of the proposed approach is described into a flowchart. A handwritten image word is taken as an input to start with. The dynamic parameters are calculated based on the input image. The measured parameters and the original input image are passed to the over segmentation module. The over segmentation module generates a set of primitives from the original input image using heuristics in conjunction with the measured parameters. From the set of primitives, a set of evaluation hypotheses are created. Each element in the evaluation contains the same image component as the original image when they combined together in order. Therefore, each

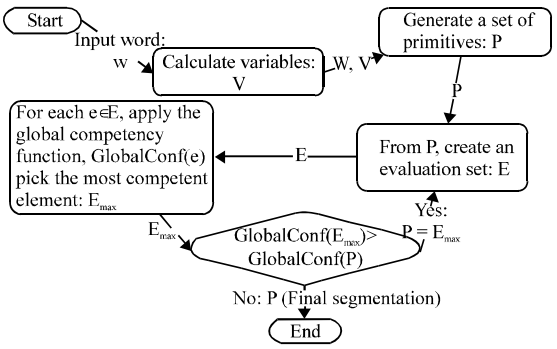

Fig. 1: The overall architecture of Binary Validation as Segmentation (BVS)

element in the evaluation set represents a complete segmentation hypothesis of the original image. The hypothesis in the evaluation set is applied with the global competency function. An element with the highest competency value represents the best segmentation hypothesis for the iteration and its competency value is compared to the beginning set of primitives which the evaluation set was generated from. If an improvement is made, the element becomes the beginning set to generate the evaluation set for the next iteration. Otherwise, the beginning set becomes the final product of segmentation. In the final segmentation product, each segment may ideally imply a single character. The great details of interim modules are described in the study.

Dynamic variables: In the proposed approach, the stroke or pen thickness is introduced as a dynamic variable. The stroke thickness is measured dynamically when the handwritten word is fed into system. Input word images are written with all different sorts of pens. Therefore, the stroke thickness will vary depending on the input images. It is more adequate to measure it dynamically rather than to assign constants to the variable. Therefore, the stroke thickness variable is context dependent.

The stroke thickness is put in use during primitives' generation by over segmentation module. The proposed approach dissects the input image into primitives which are partial characters. The binary validation builds up the characters based on the primitives. Therefore, it is a very crucial task to make sure that none of primitives contains pixel components from different characters which is called under segmentation or missed segmentation error. The status of primitive generation with no under-segmentation error is called the full over segmentation. The use of the stroke thickness variable is to ensure the full over segmentation during primitive generation process.

The stroke thickness is the most occurring continuous foreground pixel count. It is measured by

scanning the segmenting word vertically and horizontally. While scanning, the occurrences are recorded and the most occurring continuous foreground pixel count becomes the stroke thickness of the segmenting word. The details of the stroke thickness measurement are described by Lee and Verma (2008).

Full over segmentation: The core idea of introducing over segmentation into the proposed approach is to generate partial characters, primitives. Binary validation module can build up whole characters by joining the primitives. The full over segmentation is the state of the over segmentation results. In the full over segmentation, none of the primitives will share the image pixels from different characters. It is very common to have a character segmented into many partial characters. However, the aim of introducing the over segmentation is to achieve the full over segmentation.

In the proposed approach, the full over segmentation is achieved through two stages. In the first stage, horizontally disconnected pixel components are segmented first. Background or white pixel tracing algorithm (Lee and Verma, 2009) separates the disconnected components vertically. Each disconnected component may contain more than one character and they are connected by ligatures. Therefore, individual outputs from the disconnected component segmentation will be connected components. After all connected components are detected and separated; each of them is passed to the second stage of the connected component segmentation. A set of rules are applied to every connected components and the results are the primitives, partial characters. Primitives after the second stages do notcontain the pixels from the two different characters.

The rules implemented in the proposed approach are minima, maxima and enforcement. Minima are extracted from the lower contour of the upper background region. Maxima are extracted from the upper contour of the lower background region. Minima and maxima are measures to find correct character boundaries. However, it is anticipated that the minima and maxima alone cannot achieve the full over segmentation. Achieving the full over segmentation in this stage is a must. Therefore, enforcement rule is incorporated to prevent from producing under segmentation error at this stage. Enforcement rule is to locate segmentation points at a certain interval. In the proposed approach, segmentation points are forced at an interval of StrokeThickness x2.

The heuristics are expected to produce many segmentation points and not all of them are valid ones. Therefore, the final over segmentation points are filtered by hole detection and consolidation measures. The hole detection removes any segmentation points crossing the hole region. It is rare cases that two characters are touching and forming a hole. On the contrary, there are many characters featuring hole regions in English alphabets. Therefore, detecting hole regions and removing segmentation points will filter the over segmentation points much more correctly. After the filtration by hole detection, the remaining segmentation points are consolidated. Up to the point, the segmentation points located by minima, maxima and enforcement might be overlapping or close to each other. Therefore, those can be consolidated into one by applying consolidation rules (Lee and Verma, 2009). The consolidated segmentation points must be fully over segmented. A primitive is a group of pixel components bounded by two neighbouring segmentation points. Those primitives will be passed to binary validation module to build up whole characters.

Binary validation: Binary validation is iteratively applied to a set of primitives created from the over segmenting module. The primitives are more likely to be partial characters at this stage. Therefore, those partial character like primitives need to be combined with the neighboring primitives and become a whole character. In each iteration, Binary validation creates evaluation sets. Neural confidence function is applied to each of them to pick one evaluation set with the highest value. When the selected evaluation set is valued higher than the current set, it is passed as the current set for the next iteration. Otherwise, it terminates iteration and outputs the current set as the final outputs. Each final primitive should represent a whole character.

Evaluation set generation: Let P be a current set of primitives:

$$P = \{P_1, P_2, P_3 \ldots P_n\}$$

Where, n is the number of primitives in P. Considering that the +sign indicates joint between two neighboring primitives, the maximum of (n - 1) Evaluation (E) sets can be expressed as follows:

$$E_1 = \{P_1+P_2, P_3 \ldots P_n\}$$

$$E_2 = \{P_1, P_2+P_3 \ldots P_n\}$$

$$E_{n-1} = \{P_1, P_2, P_3 \ldots P_{n-1}+P_n\}$$

However, a joint between two neighboring primitives are allowed only if a joint condition is satisfied. The joint condition concerns connected component identification

and Continuous Foreground Transition (CFT) model. Recapping the over segmentation process, the input word went through the disconnected component segmentation process and the output of the process was a set of connected components. Each connected component may contain one or more connected characters. At the time, each connected component was assigned with a unique id. The primitives from the same connected component shares the connected component id. Therefore, the joint between primitives during validation is only allowed between two primitives with the connected component id.

Continuous Foreground Transition (CFT) model: CFT indicates the maximum number of horizontal black to white pixel transition for a row of pixels from a primitive. To obtain CFT from a primitive, rastering technique is used for scanning the primitive. A primitive is an image which can be represented into two dimensions, rows and columns of pixels. Therefore, there are as many CFT as the number of rows for a primitive. The highest CFT among rows represents the CFT for the primitive.

CFT model for English (EngCFT) language: As mentioned earlier, a joint condition of two neighboring primitives into one is to compare CFT of the combined primitive to a preset variable EngCFT. The use of EngCFT is to prevent from producing under-segmentation error during validation. As shown in Table 1, CFT for 52 classes of English alphabets has been generated from pre-segmented handwritten character images. The pre-segmented character images are the same data used to train two-class neural networks classifier. From the Table 1, EngCFTis assigned with 5 because the highest value of the Table 1 is 5 from M or m. Therefore, any joint between two neighboring primitives are prohibited if CFT of the combined primitive is >5. During validation, any primitives with higher CFT value than EngCFT are assumed to have more than one character. Therefore, it prevents from joining with other primitives; it makes CFT as a measure to prevent under-segmentation errors.

Table 1: Continuous Foreground Transition (CFT) for each alphabet class generated from pre-segmented character images used to train neural classifier

Class	CFT	Class	CFT	Class	CFT	Class	CFT
A	4	g	3	N	4	t	3
a	4	H	4	n	4	U	2
B	4	h	3	O	3	u	4
b	3	I	3	o	4	V	3
C	3	i	1	P	3	v	3
c	3	J	4	p	3	W	4
D	3	j	3	Q	3	w	4
d	4	K	4	q	3	X	3
E	3	k	3	R	4	x	3
e	4	L	3	r	3	Y	3
F	3	l	2	S	3	y	3
f	3	M	5	s	3	Z	3
G	4	m	5	T	4	z	2

Calculating confidence for evaluation sets: To calculate confidence value for each evaluation set, the proposed approach utilizes the neural networks outputs. The Confidence function, ConfNN(P) indicates a Class1 output value from two Class neural networks when a primitive, P is input. Class1 of two implies the likelihood of being a correct character and Class2 indicates the likelihood of being rubbish (non-character). Let GlobalConf (W) be a confidence function where W is anevaluation set of n primitives. GlobalConf (W) is sought by calculating the average:

$$W = \{P_1, P_2, P_3 \text{ and } P_n\}$$
$$\text{GlobalConf}(W) = \frac{\sum_{k=1}^{n} \text{ConfNN}(P_k)}{n} \quad (1)$$

Therefore, an evaluation set with the highest confidence value, E_{max} can be obtained by:

$$E_{max} = \left(e \in E^{\text{ArgMax}}\, \text{GlobalConf}(e)\right) \quad (2)$$

where, e is an element of evaluation sets, E.

Continuous Foreground Transition Matrix (CFTM): In the study, CFT and EngCFT are introduced to prevent under-segmentation errors during validation. In this study, the use of CFTM is explained to boost the competency of primitives by being multiplied to neural network outputs of primitives inside evaluation function. The ideal outcome of introducing this feature is to reduce overall over segmentation and bad segmentation errors. The method to measure CFT of a primitive is the same as explained previously. However, the distinguishing difference of CFTM is that it is measured in both directions; vertical as well as horizontal by the way, CFT was measured in horizontal direction only. CFT was expressed as a single digit which representing horizontal count. However, CFTM is expressed in double digit format. For example, 34 is a value of CFTM. The 3 is for the horizontal count and the 4 for the vertical count of a primitive.

To model the pattern of CFTM for English language, it requires the pre segmented character images. Therefore, the same database was used to model CFTM for English. Similar to obtaining EngCFT from the database, CFTM for each sample is acquired and recorded. The pattern of

Table 2: It shows global CFTM across the 52 classes. The patterns of CFTM have been extracted, populated and normalized

Pattern	Population	Normalized	Pattern	Population	Normalized
44	5	0.0032	12	36	0.0231
22	432	0.2769	21	29	0.0186
35	3	0.0019	43	41	0.0263
23	358	0.2295	42	59	0.0378
36	3	0.0019	41	3	0.0019
33	232	0.1487	32	171	0.1096
24	42	0.0269	31	18	0.0115
34	53	0.0340	51	1	0.0006
25	12	0.0077	52	3	0.0019
13	12	0.0077	53	5	0.0032
14	1	0.0006	63	1	0.0006
11	38	0.0244	54	2	0.0013

CFTM and the number of samples per pattern for each class have been extracted. After obtaining the patterns of CFTM for each class, the global CFTM can be repopulated in the order of CFTM patterns. The global population and normalized value for each CFTM pattern has been shown in Table 2. For example, the pattern of CFTM 44 has 4 samples and its normalized value is 0.0032. The pattern and its normalized value of a primitive can be incorporated into evaluation function in Eq. 1. Let M (P) be a CFTM function when P is a primitive:

$$\text{GlobalConf}(W)\frac{\sum_{k=1}^{n}\{\text{ConfNN}(P_k)\times M(P_k)\}}{n}$$

Binary validation terminating conditions: The improvement is made when the confidence value of an evaluation set with the highest confidence value (GlobalConf(E_{max})) is greater than the confidence value (GlobalConf(S)) of the current set S. Then, the evaluation set is promoted as a current set for next iteration.

Let E_{max} and P be an evaluation set with the highest confidence value and a current searching set for the iteration, respectively. An improvement is made: If

$$\text{GlobalConf}(E_{max})>\text{GlobalConf}(P)$$

Then:

$$P = E_{max}$$

Binary validation will be repeated until certain conditions are erupted. There are two terminating conditions enforced. One of them occurs when there is no evaluation set to be created from the current set. No evaluation set will be produced when the current set has only single primitive or when joints are denied due to the joint condition. The other enforces binary validation to terminate when there is no improvement. It ensures that the current set would be the best combinations of primitives.

Binary validation in steps:

- Line 1: Create primitives from input word
- Line 2: Create evaluation sets from the current set
- Line 3: Find the most competent evaluation set
- Line 4: Decide whether the competency of the selected evaluation set is improved over the current set
- Line 5: If improved, assign the selected evaluation set to the current set. Go to line 2
- Line 6: Output the current set as final primitives

EXPERIMENTS AND RESULTS

Implementation: In the proposed system, all algorithms have been implemented in Java programming language using object oriented principles.

Database preparation: The experiment is conducted on a CEDAR benchmark database to perform the comparative analysis against the segmentation results from the literature. The final segmentation was experimented on 317 words from CEDAR\TEST\CITIES\BD directory. Preprocessing is a vital step in handwriting recognition.

However, the outcomes of incorporating preprocessing are not always positive. Some favors preprocessing elements such as noise removal, slant or slope correct and binarization, etc. On the other hand, others argue that the preprocessing might cause loss of vital information by removing or altering the originality. In the proposed approach, slant or slope preprocessing element could have improved the final segmentation results. In fact, almost every word in CEDAR database used for experiments has very fluctuating slant and slope angle. Therefore, inclusion of skew correction might be very prospective. Therefore, the 317 words were pre-processed in advance following the principles by Vinciarelli and Luettin (2001). During the pre-processing only slant and slope angles are corrected.

Parameters setup: During the experiments, a static parameter was tuned into a constant. In this study, there was a single dynamic parameter, Stroke Thickness (ST). ST was context-dependant and measured automatically before the segmentation by inspecting the segmenting word image. The ST was used during creating primitives in the study. The main reason to include the over segmentation module was achieving the full over segmentation. However, the heuristics of minima and maxima alone did not achieve the full over segmentation. Therefore, enforcement rule was supplemented to achieve the full over segmentation. The enforcement was to locate segmentation points at a certain interval. The interval was calculated by ST multiplied by two. There was no under segmentation error in the over segmentation results which means achieving the full over segmentation.

Table 3: Over segmentation performance by heuristic rules

Heuristic rules applied	Missed segmentation boundary (%)
Minima and maxima	8.79
Minima, maxima and enforcement	0.00

Over segmentation performance: The primary purpose of over segmentation is to generate a complete set of segmentation points. The complete set must contain all correct letter boundaries plus it may contain excessive incorrect segmentation boundaries which is indicated as full over segmentation. In the proposed approach, there were three rules employed to achieve full over segmentation. The three rules are based on minima, maxima and enforcement. The minima and maxima intended to find correct segmentation boundaries. However, minima and maxima alone do not guarantee the full over segmentation. Enforcement guarantees the full over segmentation. The over segmentation performance by heuristic rules is shown in Table 3. The statistics are obtained by applying minima and maxima to 317 words described in the study and the missed segmentation points between letters are counted by manual inspection. The total number of missed segmentation points across the 317 words was divided by the total number of characters in the 317 words. The minima and maxima algorithms alone did not achieve full over segmentation because they missed to find 8.79% correct segmentation boundaries. Therefore, enforcement was added along with minima and maxima and over segmentation was performed again. When the three rules are applied together, there was no missed segmentation point. Hence, it concludes that the over segmentation was successful.

Neural networks classifier: A MLP neural network with a single hidden layer was trained on pre segmented characters with back propagation learning algorithm. It takes 100 inputs and produces 2 outputs. The two outputs represent Class1 and Class2. Class1 is likelihood of being correct character and Class2 is of being rubbish character. Training data for Class1 was pre segmented characters from CEDAR/TRAIN/BINANUM directory. The number of training samples for Class1 includes 1560 pre segmented character images. In English, there are 52 alphabet letters. Therefore, 1560 images consist of 30 images per alphabet letter. Training data for class 2 was generated by over segmenting words from CEDAR/TRAIN/CITY directory. From the over segmentation, 1560 rubbish look-like primitives were manually selected for Class2 training data to balance with Class1 training data. Using the first fold of the tenfold cross-validation, the number of hidden units and the number of iteration were incremented by an interval for training. With 13 hidden units at 4000 iterations, the neural classifier was trained optimally to achieve around 78.85% classification accuracy at most and used in the experiments.

Table 4: Tenfold cross validation of the neural network classifier with 13 hidden units at 4000 iterations

			Accuracy (%)	
Folds	No. of training samples	No. of test samples	Training samples	Test samples
1	2808	312	89.53	78.85
2	-	-	85.75	75.64
3	-	-	87.11	78.53
4	-	-	92.49	76.92
5	-	-	92.52	76.28
6	-	-	87.29	69.87
7	-	-	83.30	68.59
8	-	-	81.52	63.14
9	-	-	87.54	63.78
10	-	-	86.86	74.36
Average	-	-	87.39	72.60

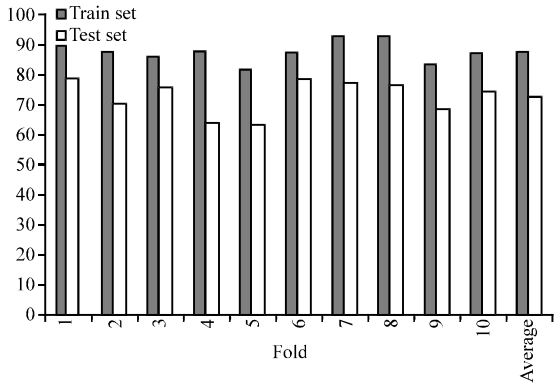

Fig. 2: Cross-validation results for neural networks classifier

Neural network training: For the better generalization of the neural classifier on given training data, ten fold cross validation was performed. The training dataset includes 3120 samples. For Class1 training samples, 1560 samples were randomly partitioned into ten. For Class0 training samples, there are 52 distinct figures; resembling 52 alphabets. In Class0 training samples, there are 30 pre segmented characters per alphabet characters. Therefore, each fold for Class0 samples contains 3 pre segmented characters per alphabet. In this study, the first fold dataset was used to find the optimal parameter settings. For the first fold, the optimal parameter settings were 13 hidden units at 4000 iterations. The maximum accuracy was 78.85 on the first fold. In the experiment, the first fold neural network model was used as the classifier since, it has the highest accuracy on the test samples (Table 4 and Fig. 2).

Use of CFT and EngCFT: The primary use of CFT is to prevent re-introducing under-segmentation errors during

Table 5: The effectiveness of using EngCFT strategy to prevent from re-introducing under segmentation errors during validation

		No. of correct prevention		No. of incorrect prevention	
EngCFT	No. of total prevention	Count	Ratio (%)	Count	Ratio (%)
5	36	35	97.22	1	2.88

validation. CFT and EngCFT are the working pair to reduce under segmentation errors. For examples, two neighboring primitives are combined into a single primitive. CFT of the combined primitive is extracted. If the CFT is greater than EngCFT, the combination is not allowed because it is regarded as containing more than a single character. To measure the effectiveness of the CFT strategy, two criteria were used. The first one is to count the number of occurrences of prevention. The second one is to inspect if the prevention is correct or not. Prevention is defined as correct if the two neighboring primitives belong to different character segments. On the other hand, prevention is regarded as incorrect if the two neighboring primitives belong to a same character segment.

The experiment was conducted by applying the proposed approach to the 311 words described earlier in database section. The variable Eng CFT is assigned with the maximum CFT from Table 1. While segmenting 311 words individually, the number of occurrences of prevention was recorded. The prevented primitives were manually inspected to check if the preventions were correct or not.

In Table 5, total 36 times of joints were prevented by the use of CFT. The 35 out 36 were correct prevention and only one joint between two neighboring primitives from a same character segment was prevented. In other words, 35 under segmentation errors were prevented during validation by Eng CFT strategy. The correct prevention rate is around 97% and incorrect prevention rate is around 3%.

Segmentation performance: One of the segmentation accuracy measurements is to check the segmentation errors. Counting the segmentation errors was traditionally done by manual inspection. However, manual inspection by the involving authors to this research may incur biased results. Moreover, there are no such rule of thumb applications or tools to inspect the correctness of segmentation. If there is this research would be vain in the first place. To minimize the bias to the segmentation result, therefore, blind inspection was performed.

The blind inspection is to pick a person unrelated to the research and the rules of over, under and bad segmentations were taught as followings. Achieving higher segmentation accuracy means lowering

Table 6: Final segmentation results of the proposed approach

			Segmentation error (%)			
Experiments	Classifier	No. of words	Under	Over	Bad	Average
BVS without CFTM	78.85%	317	1.96	11.55	7.33	6.94
BVSwith CFTM	-	-	1.96	2.76	5.32	3.35

segmentation errors. The segmentation errors are categorized into three types such as under segmentation, over segmentation and bad segmentation errors (Yanikoglu and Sandon, 1998). The under segmentation error is defined if segmentation boundary is missed between two neighboring characters. The over segmentation error happens if a character segment is divided into more than two primitives exclusively. The bad segmentation error is defined as incorrect segmentation boundary which does not belong to under segmentation or over segmentation errors.

Two experiments were conducted. The first experiment was to applying the BVS with Eq. 1 individually to the 317 CEDAR words described in database section. The second experiment was to applying the BVS with Eq. 3 to the same database. From the final segmentation output of each word, the segmentation errors were accumulated by manual blind inspection. For the both experiments, the parameter, EngCFT was statically assigned with the maximum value 5 from Table 1. The figures for the segmentation errors in Table 6 are calculated by dividing each categorical accumulated error by the total number of characters in 317 words. In the first experiment, the highest segmentation error was caused by over segmentation followed by bad segmentation and under segmentation in order. The overall average error rate was 6.94% for the first experiment. From the second experiment, the highest error was caused by bad segmentation error, followed by over segmentation and under segmentation, respectively. The overall average error from the second experiment was 3.35%.

Segmentation performance and neural classifier accuracy: To generalize the relationship between segmentation performance and neural classifier accuracies, three combinations of BVSCFTM and neural classifiers were experimented on the CEDAR database. The best classifier model has 78.8% classification accuracy and it is the same one as used in Table 6. The second best has 62.9% classification accuracy. The lowest classifier model is 41.1% accuracy. As shown in Table 7, the average segmentation error for the classifier with 78.8% accuracy is 3.35%. Also, the average classification errors for 62.9 and 41.1% classifiers are 5.11 and 7.21% accordingly. The result anticipates the higher

Table 7: Segmentation performance depending on different classifier accuracies

Methodology	Classifier accuracy (%)	Database	Segmentation errors (%) Over	Under	Bad	Average
BVSCFTM	78.8	CEDAR 317	2.76	1.96	5.32	3.35
	62.9		2.76	4.86	7.72	5.11
	41.1		5.66	5.46	10.62	7.21

Table 8: Character classification rate from the final segmentation results and word recognition rate based on correctly classified characters on 317 lexical words

Classifier accuracy for 53 classes (%)	Character classification rate (%)	Word recognition rate (%)
81.73	50.27	80.12

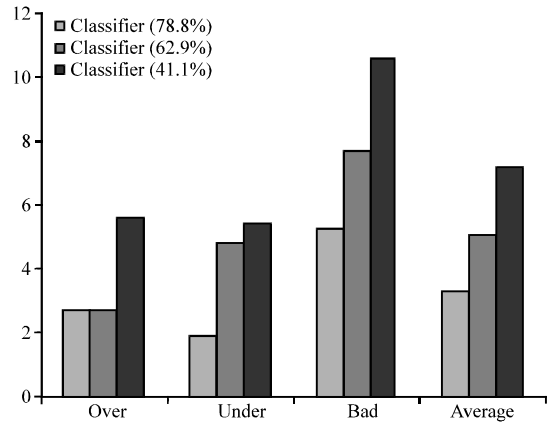

Fig. 3: Segmentation performance on different classification accuracies

average segmentation errors on the less accurate classifiers. The graphical representation of the results is shown in Fig. 3.

Character classification and word recognition: One of the main purposes of introducing segmentation step is to extract individual characters and to classify them individually without context. Based on the correctly classified characters, words can be recognized. Therefore, higher character classification rate would increase higher word recognition rate. To measure the character classification, the final segments for each input word are fed into the neural classifier which was used during segmentation step. The numbers of correctly classified characters are accumulated across the input words. The final character classification rate is calculated by obtaining the percentage of the accumulated correct classifications against the total number of characters. As shown in Table 8, the character classification rate from the experiment was 50.27%. Word recognition is performed based on the correctly classified characters. As seen in the character classification results above, more than half characters of each input word were recognized correctly. To recognize a word from a set of characters, each lexical word was tested for matching against those characters and a lexical word with highest matching competency becomes a recognized word. As shown in Table 8, the word recognition rate was 80.12% on 317 lexical words.

As shown in Table 6, it is obvious that the 1.96% of under segmentation error was re-introduced by the proposed binary validation by combining two neighboring primitives which belongs to the different character segment. However, the under segmentation error is very minor compared to over or bad in both experiments. Especially, over segmentation error in the first experiment was the highest. The over and bad segmentation errors during the proposed binary validation can be contributed by getting incorrect confidence value of primitives from neural classifier. The proposed neural binary validation entirely depends on the confidence value from neural classifier apart from the interruption by CFT strategy. Theemployed neural classifier was trained up to around 79% classification accuracy. However, it was very challenging to model the rubbish class which virtually has no pattern. Therefore, rubbish look-like patterns could be unknown to the classifier because they weren't included in the training data. Those unknown pattern could give incorrect output values.

The use of CFT and EngCFT was investigated to find out the effectiveness of under segmentation error prevention. The performance was measured by counting the number of correct and incorrect prevention. Through the experiment, 36 preventions were discovered; 35 preventions were correct and 1 was incorrect. Therefore, the use of CFT and EngCFT was very effective to prevent under segmentation errors.

In the proposed approach, CFTM was modeled from pre segmented characters and the use of CFTM was injected into evaluation function during validation. The ideal goal of its use was to reduce the overall segmentation errors to improve the segmentation accuracy. To investigate the objective, two different segmentation experiments were conducted on the same database. The first experiment was to apply BVS without CFTM. The second one was to use BVS with CFTM. The overall segmentation error from the first experiment was 6.94%. The second experiment showed 3.35% of overall segmentation error. Through the experiment results and analysis, the use of CFTM was very effective to reduce the segmentation errors from 6.94-3.35%. The comparison of the segmentation performance is shown in Fig. 4.

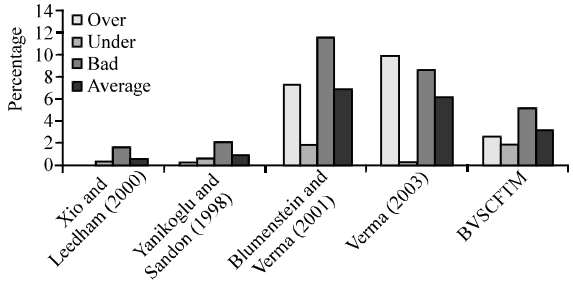

Fig. 4: Visualization of the comparative analysis

Table 9: Comparing the final segmentation performance of the proposed approach (BVS) to the existing segmentation methodologies in the literature

Methodology in the literature	Database		Segmentation error rate (%) Over	Under	Bad	Average
Xiao and Leedham (2000)	CEDAR	200	0.00	0.50	1.80	0.76
Yanikoglu and Sandon (1998)	Local	750	0.10	0.80	2.30	1.06
Blumenstein and Verma (2001)	CEDAR	317	7.40	2.00	11.60	7.00
Verm (2003)	CEDAR	317	10.00	0.20	8.70	6.30
BVSCFTM	CEDAR	317	2.76	1.96	5.32	3.35

The final segmentation performance of the proposed approach has been compared to the existing ones from the literature and shown in Table 9. The over segmentation error rate of the proposed approach was lower than Verma (2003) and Blumenstein and Verma (2001) but higher than Yanikoglu and Sandon (1998) and Xiao and Leedham (2000). The under segmentation error of the proposed approach was higher than Verma (2003), Xiao and Leedham (2000) and Yanikoglu and Sandon (1998). The bad segmentation error of the proposed approach was lower than Verma (2003) and Blumenstein and Verma (2001) but higher than Xiao and Leedham (2000) and Yanikoglu and Sandon (1998). Finally, the average error rate of the proposed approach is lower than the approach presented by Blumenstein and Verma (2001) and Verma (2003) but higher than the approach presented by Xiao and Leedham (2000) and Yanikoglu and Sandon (1998). However, comparing the result of Xiao and Leedham (2000) and Yanikoglu and Sandon (1998) is unfair because researchers Xiao and Leedham (2000) used only 200 words by excluding the distorted words from the same benchmark database. In addition, approach by Verma (2003) and Blumenstein and Verma (2001) uses over segmentation and validation strategies similar to the proposed BVS. Therefore, it could be the fair comparison of the result of Verma (2003) and Blumenstein and Verma (2001) and the proposed approach. As shown in the result, the proposed segmentation strategy is promising and better compared to Verma (2003) and

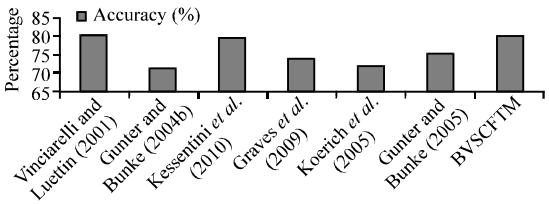

Fig. 5: Graph representation of the word recognition accuracy

Table 10: Word recognition rate comparison

Method	Database	Words	Accuracy (%)
Vinciarelli and Luettin (2001)	Cambridge	1020	80.37
Gunter and Bunke (2004a, b)	IAM	1066	71.58
Kessentini *et al.* (2010)	IFN/ENIT	6033	79.60
Graves *et al.* (2009)	IAM	20000	74.10
Koerich *et al.* (2005)	SRTP	4674	72.39
Gunter and Bunke (2005)	IAM	2296	75.60
BVSCFTM	CEDAR	317	80.12

Blumenstein and Verma (2001) in terms of average segmentation error. Based on the character classification shown in Table 8, the word recognition was produced by confidence-based string matching algorithm. The word recognition rate is compared to the published recognition results in Table 10. The proposed algorithm performed better than Gunter and Bunke (2004a, b), Kessentini *et al.* (2010), Graves *et al.* (2009), Koerich *et al.* (2005) and Gunter and Bunke (2005) and very close to the performance of Vinciarelli and Luettin (2001). The comparison of the word recognition performance is shown in Fig. 5.

CONCLUSION

In this study, a novel Binary Validation as Segmentation (BVS) approach has been proposed as a segmentation strategy for off line cursive handwriting recognition. The experiments using the proposed approach on CEDAR benchmark dataset have been conducted.

The proposed approach builds up the whole characters from the primitives. The primitives are generated from the over segmentation module. Introducing the over segmentation module was to achieve the full over segmentation. Fully over segmented primitives do not share pixel components from different characters. Through the experiments, the combination of minima, maxima and enforcement produced the full over segmentation. Without enforcement however, the full over segmentation was not achieved.

Continuous Foreground Transition (CFT) and EngCFT worked together to prevent from reproducing the

under segmentation error. The fidelity of the prevention success was around 9%. Also the proposed approach was experimented with different level of classifiers. From the experiment, the segmentation error rate was decrease while the classification accuracy increases.

The over, under and bad segmentation errors were 2.76, 1.96 and 5.32%, respectively. A comparative analysis showed that the proposed approach performed well in comparison to some existing techniques in the literature. The analysis also showed that the validation function with Continuous Foreground Transition Matrix (CFTM) has played a major role in reducing the over segmentation and bad segmentation errors.

Based on the final segmentation results, the character classification was tested. Abiut >50% of primitives are correctly classified with around 81% of 53 class classifier. Based on the characters classification results, simple string matching algorithm was used to find the feasible word. The word recognition rate was over 80%. The word recognition rate was very competitive compared to the methods from the literature.

RECOMMENDATION

As future research, classifier can be trained better to achieve higher accuracy. With higher level classifier, the segmentation performance can be improved. Also word recognition can be improved by employing the multiple experts.

REFERENCES

Al-Hamad, H.A. and R.A. Zitar, 2010. Development of an efficient neural-based segmentation technique for Arabic handwriting recognition. Pattern Recogn., 43: 2773-2798.

Arica, N. and F. Yarman-Vural, 2002. Optical character recognition for cursive handwriting. IEEE Trans. Patt. Anal. Mach. Intell., 24: 801-813.

Arica, N. and F.T. Yarman-Vural, 2001. An overview of character recognition focused on off-line handwriting. IEEE Trans. Syst. Man Cybernet. C Appl. Rev., 31: 216-233.

Benouareth, A., A. Ennaji and M. Sellami, 2008. Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition. Patt. Recog. Lett., 29: 1742-1752.

Bertolami, R., M. Zimmermann and H. Bunke, 2006. Rejection strategies for offline handwritten text line recognition. Patt. Recog. Lett., 27: 2005-2012.

Blumenstein, M. and B. Verma, 2001. Analysis of segmentation performance on the CEDAR benchmark database. Proceedings of the 6th International Conference on Document Analysis and Recognition, Sept. 10-13, Seattle, USA., pp: 1142-1146.

Casey, R.G. and E. Lecolinent, 1996. A survey of methods and strategies in character segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 18: 690-706.

Elnagar, A. and R. Alhajj, 2003. Segmentation of connected handwritten numeral strings. Patt. Recog., 36: 625-634.

Fujisawa, H., 2008. Forty years of research in character and document recognition: An industrial perspective. Patt. Recog., 41: 2435-2446.

Graves, A., M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke and J. Schmidhuber, 2009. A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Patt. Anal. Mach. Intell., 31: 855-868.

Gunter, S. and H. Bunke, 2004a. Feature selection algorithms for the generation of multiple classifier systems and their application to handwritten word recognition. Patt. Recog. Lett., 25: 1323-1336.

Gunter, S. and H. Bunke, 2004b. HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. Patt. Recog., 37: 2069-2079.

Gunter, S. and H. Bunke, 2005. Off-line cursive handwriting recognition using multiple classifier systems-on the influence of vocabulary, ensemble and training set size. Optics Lasers Eng., 43: 437-454.

Hussein, K.M., A. Agarwal, A. Gupta and P.S.P. Wang, 1999. A knowledge-based segmentation algorithm for enhanced recognition of handwritten courtesy amounts. Patt. Recog., 32: 305-316.

Kessentini, Y., T. Paquet and A.M.B. Hamadou, 2010. Off-line handwritten word recognition using multi-stream hidden Markov models. Patt. Recog. Lett., 31: 60-70.

Kim, K.K., J.H. Kim and C.Y. Suen, 2002. Segmentation-based recognition of handwritten touching pairs of digits using structural features. Patt. Recog. Lett., 23: 13-24.

Koerich, A.L., R. Sabourin and C.Y. Suen, 2005. Recognition and verification of unconstrained handwritten words. IEEE Trans. Patt. Anal. Mach. Intell., 27: 1509-1522.

Lee, H. and B. Verma, 2008. A novel multiple experts and fusion based segmentation algorithm for cursive handwriting recognition. Proceedings of the IEEE International Joint Conference on Neural Networks, (IEEE World Congress on Computational Intelligence), June 1-8, Hong Kong, pp: 2994-2999.

Lee, H. and B. Verma, 2009. Binary segmentation with neural validation for cursive handwriting recognition. Proceedings of the International Joint Conference on Neural Networks, June 14-19, Atlanta, GA, pp: 1730-1735.

Liu, J. and P. Gader, 2002. Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. Patt. Recog., 35: 2061-2071.

Mozaffari, S., K. Faez, V. Margner and H. El-Abed, 2008. Lexicon reduction using dots for off-line Farsi/Arabic handwritten word recognition. Patt. Recog. Lett., 29: 724-734.

Nopsuwanchai, R., A. Biem and W.F. Clocksin, 2006. Maximization of mutual information for offline Thai handwriting recognition. IEEE Trans. Patt. Anal. Mach. Intell., 28: 1347-1351.

Sadri, J., C.Y. Suen and T.D. Bui, 2007. A genetic framework using contextual knowledge for segmentation and recognition of handwritten numeral strings. Patt. Recog., 40: 898-919.

Su, T.H., T.W. Zhang, D.J. Guan and H.J. Huang, 2009. Off-line recognition of realistic Chinese handwriting using segmentation-free strategy. Patt. Recog., 42: 167-182.

Suresh, R.M. and S. Arumugam, 2007. Fuzzy technique based recognition of handwritten characters. Image Vision Comput., 25: 230-239.

Vellasques, E., L.S. Oliveira, A.S. Britto Jr., A.L. Koerich and R. Sabourin, 2008. Filtering segmentation cuts for digit string recognition. Patt. Recog., 41: 3044-3053.

Verma, B., 2003. A contour code feature based segmentation for handwriting recognition. Proceedings of the 7th International Conference on Document Analysis and Recognition, Aug. 3-6, Edinburgh, Scotland, pp: 1203-1207.

Verma, B., M. Blumenstein and M. Ghosh, 2004. A novel approach for structural feature extraction: Contour vs. direction. Patt. Recog. Lett., 25: 975-988.

Verma, B., P. Gader and W. Chen, 2001. Fusion of multiple handwritten word recognition techniques. Patt. Recog. Lett., 22: 991-998.

Viard-Gaudin, C., P.M. Lallican and S. Knerr, 2005. Recognition-directed recovering of temporal information from handwriting images. Patt. Recog. Lett., 26: 2537-2548.

Vinciarelli, A. and J. Luettin, 2001. A new normalization technique for cursive handwritten words. Patt. Recog. Lett., 22: 1043-1050.

Vinciarelli, A., 2005. Application of information retrieval techniques to single writer documents. Patt. Recog. Lett., 26: 2262-2271.

Xiao, X. and G. Leedham, 2000. Knowledge-based English cursive script segmentation. Patt. Recog. Lett., 21: 945-954.

Xu, Q., L. Lam and C. Suen, 2003. Automatic segmentation and recognition system for handwritten dates on Canadian bank cheques. Proceedings of the 7th International Conference on Document Analysis and Recognition, Aug. 3-6, Edinburgh, Scotland, pp: 704-708.

Yanikoglu, B. and P.A. Sandon, 1998. Segmentation of off-line cursive handwriting using linear programming. Patt. Recog., 31: 1825-1833.

Zhao, S., Z. Chi, P. Shi and H. Yan, 2003. Two-stage segmentation of unconstrained handwritten Chinese characters. Patt. Recog., 36: 145-156.