An Artificial Neural Network Approach for User Class-Dependent Off-Line Sentence Segmentation

César A. M. Carvalho and George D. C. Cavalcanti

Abstract— **In this paper, we present an Artificial Neural Network (ANN) architecture for segmenting unconstrained handwritten sentences in the English language into single words. Feature extraction is performed on a line of text to feed an ANN that classifies each column image as belonging to a word or gap between words. Thus, a sequence of columns of the same class represents words and inter-word gaps. Through experimentation, which was performed using the IAM database, it was determined that the proposed approach achieved better results than the traditional Gap Metric approach for handwriting sentence segmentation.**

I. Introduction

THE automatic recognition of handwritten texts is a challenging task with important commercial applications, such as bank system processing, mail system processing for reading addresses and postal codes and systems for historical document indexation. In the academic environment, there is an endeavor to improve the accuracy rate and time performance of this task in a large number of application fields [1][3][4].

Automatic text segmentation is one of the initial steps leading to the complete recognition of handwritten sentences in systems that appraise words separately. Therefore, a good performance in terms of accuracy rate is essential, as sentences that were uncorrectly segmented require manual intervention, which is much more expensive.

The task of obtaining words from a machine-printed text is simpler than from a handwritten text because the spacing between characters and words are regular in machine-printed texts and the gaps are easily estimated. Handwritten texts, however, are not uniform and therefore represent a more difficult, elaborate task. Difficulties in handwritten sentence segmentation included irregular distances, variation in character size, inclination in the writing, noise, the influence of document background and blurring.

Most segmentation methods consider spaces between words to be larger than those between characters. Seni and Cohen [1] presented eight different methods for distance calculation between components: Bounding Box, Euclidean, Run-Length distances and others that use heuristics. The best accuracy rate achieved was 90.30%, using the Run-length approach plus an heuristic plan. Mahadevan and Nagabushnam [2] proposed a technique based on distances between Convex Hulls to estimate the gap size between characters and words. The Convex Hull method achieved better results (93.30% accuracy rate) than the methods introduced by Seni and Cohen. Both experiments were performed on the same database, composed of street lines, city/state/ZIP lines and personal name lines extracted from United States postal address images [2].

More recently, Marti and Bunke [3] and Manmatha and Rothfeder [4] tested the Convex Hull method on full-page handwritten text extracted from the public IAM database [8]. Their experiment achieved 95.56% and 94.40% accuracy rates, respectively. Other methods, such as the Hidden Markov Model and Artificial Neural Networks (ANN) [5], can be used to perform sentence segmentation based on an iterative segmentation/recognition process. With such methods, the image is divided into smaller images that are submitted to a recognition module, which indicates whether the image was recognized as a known word. This procedure is repeated until reaching a stopping criterion. However, this approach has a clear drawback - it is bound by a limited vocabulary of words.

This paper addresses the problem of unconstrained sentence segmentation based on Artificial Neural Network. The method created seeks to overcome the following difficulties: i) The segmentation system based on Gap Metrics needs heuristics to optimize and adapt it to different tasks [6]; ii) The HMM-MLP approach, presented in [5], has vocabulary limitation. Our method was evaluated using the IAM database. The experiments revealed promising results, achieving better error rates than traditional methods.

The structure of this paper is as follows: Section 2 details the ANN segmentation method. Section 3 discusses the experiments and presents the results of the ANN method versus Convex Hull Gap Metrics. Section 4 presents the final considerations on the present work.

II. Artificial Neural Network Approach For Sentence Segmentation

The handwritten text line segmentation method present in this paper is based on Artificial Neural Networks. We have used a Multi-Layer Percetron (MLP) trained with a resilient backpropagation (RPROP) learning algorithm.

Gap Metrics segmentation methods are based on distances between image components (connected components or convex hulls). Segmentation consists of determining a threshold value that separates which distances are intra-words and which are inter-words. The ANN segmentation method used

This work was supported in part by the Brazilian National Research Council CNPq (Proc. 478534/2006-0).

Authors are with Center of Informatics (CIn), Federal University of Pernambuco (UFPE), P.O. Box 7851, Cidade Universitria, Cep: 50.740-530 Recife PE Brazil (corresponding authors to provide phone:55-81-21268430; e-mails: camc@cin.ufpe.br; gdcc@cin.ufpe.br)

in this paper classifies a set of features as a word or a space between words.

One difficulty that emerges when using ANNs with images is how to achieve a representative set of features to be inserted as the input of the classifier. We decided to use nine geometrical quantities, based on Marti and Bunke's paper [7], calculated over a sliding window of one column width and the height of the image. These characteristics are acquired from left to right on each handwritten text line column. The input image is then represented by a sequence of feature vectors with 9 dimensions versus image width (Figure 1).

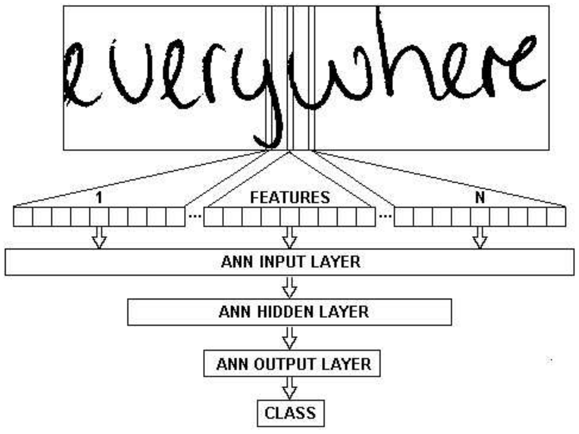

Fig. 1. Sliding Window Architecture

The nine features extracted from each window are explained as follows:

1) Window weight: total number of black pixels.

$$f_1 = \sum_{y=1}^{m} p(x,y),$$

where p is the pixel value (0 or 1) and m is the image height

2) Center of gravity.

$$f_2 = \frac{1}{m}\sum_{y=1}^{m} y \cdot p(x,y)$$

3) Second order moment.

$$f_3 = \frac{1}{m^2}\sum_{y=1}^{m} y^2 \cdot p(x,y)$$

4) Position of the upper contour: coordinate of the highest window pixel.
5) Position of the lower contour: coordinate of the lowest window pixel.
6) Gradient of the upper contour: direction (up, straight or down) acquired comparing the position of the upper contour of the previous column and current column.
7) Gradient of the lower contour: direction (up, straight or down) acquired comparing the position of the lower contour of the previous column and current column.
8) Black-white transitions: total number of black-white transitions observed in up to down direction.
9) Black pixels between the upper and lower contours.

The input of the system is represented by the handwritten text line images. We have not used any kind of normalization (such as skew, slant or writing width). Therefore, some of the nine features presented in [7] were modified in an attempt to equalize the influence of each feature over the classification. Basically, the modification was the addition of a normalization factor:

- Features 1, 4, 5 and 9: the normalization factor is 1/(image height);
- Feature 2 and 3: the normalization factor is the 1/(maximum value that each formula can reach). This value occurs when all pixels of the image column are black.

Only two classes are needed for the segmentation problem designed in this paper. Class 0 represents the intra-word columns and class 1 the inter-word column.

A. System Overview

This section details the system phases.

1) Pattern composition: A flowchart representing the first system phase is illustrated in Figure 2. Initially, the system receives images from the text lines as input and executes "Feature Extraction". Each image column is then represented by nine features. The next step is to generate the expected classification for each column ("Column Classification"). The columns in which the coordinates belong to a word are classified as Class 0. Otherwise, columns are classified as Class 1. Column classification can be performed automatically, because we have used handwritten text line images from the IAM Database 3.0 [8]. This database has meta-information on the lines that describes the Bounding Boxes of words in the handwritten text lines. "Pattern Generation" consists of joining the nine features to their respective classification in order to create a pattern for each column.

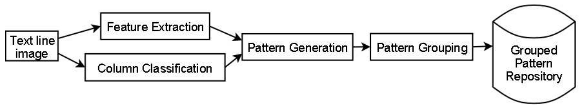

Fig. 2. Pattern composition

It is difficult to classify a pattern as belonging to a word or a gap without analyzing its neighbors. Thus, "Pattern Grouping" was developed to improve the ANN classification performance. A pattern is originally composed of nine features and one class identifier. After the grouping process for N patterns, a pattern will have: $N * 9$ features and one class identifier. The created pattern classification is the same as the original inner pattern.

Table I displays a size-three pattern grouping ($N = 3$). In the first line, there are seven patterns with their feature set and respective class. F_i is the representation of an image column by its 9 features. After the grouping process, five patterns are created (line two) with the feature set composed of the features of the three original patterns. The classification of the new pattern corresponds to the original inner pattern.

TABLE I
SIZE-THREE PATTERN GROUPING

Original pattern	F1	F2	F3	F4	F5	F6	F7
Pattern class	0	0	0	1	0	0	0
Created Pattern		F1F2F3	F2F3F4	F3F4F5	F4F5F6	F5F6F7	
Pattern class		0	0	1	0	0	

The Grouped Pattern Repository (Figure 2), stores all patterns that will be used in the ANN training and test phases.

2) ANN Training and Test: The second phase of the system is illustrated by the flowchart in Figure 3.

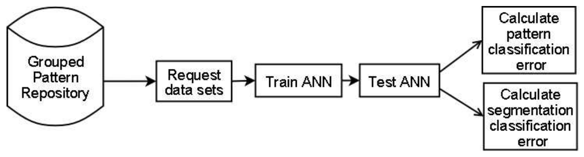
Grouped Pattern Repository
Request data sets
Train ANN
Test ANN
Calculate pattern classification error
Calculate segmentation classification error

Fig. 3. ANN Training and Test

In this stage, three pattern sets are retrieved from the repository (Request data sets): training, validation and test sets. The patterns created from a single image must be grouped and ordered into a unique data set. The ANN is trained with the two former pattern sets (Train ANN), and then an evaluation is performed, classifying the patterns of the test set (Test ANN).

Two kinds of errors are calculated in the test set classification: i) The Pattern classification error refers to the percentage of wrongly classified patterns; ii) The Segmentation classification error considers the number of wrongly classified runs. A run is a pattern sequence that has the same classification (belonging to the same class). The pattern classification error has no relevance to our work, as the segmentation error considers a sequence of patterns rather than an isolated pattern. Thus, one or more wrongly classified patterns in a single word is considered an single segmentation error.

Considering the pattern classification of the supposed text line in Figure 4, it is possible to exemplify how the Segmentation classification error is calculated. There are five runs, three words (Class 0) and two gaps between words (Class 1). The ANN Classification of the pattern failed in two runs - the first and third. Thus, the segmentation error is 2/5 or 40%.

In Figure 4, no margin of error in the word boundary was considered. However, if we adopt one pixel as error tolerance, then a single wrongly classified pattern localized in the word boundary is not considered as belonging to the word and, consequently, the segmentation error rate is not increased.

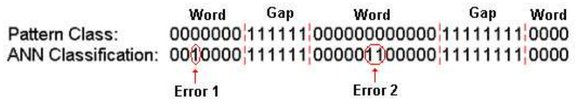
Word
Gap
Word
Gap
Word
Pattern Class: 0000000 111111 000000000000 11111111 0000
ANN Classification: 0010000 111111 000001100000 11111111 0000
Error 1
Error 2

Fig. 4. Segmentation error

III. EXPERIMENTS AND RESULTS

Like Marti and Bunke [3] and Manmatha and Rothfeder [4], our experiments were performed using the IAM-database [8]. This database contains forms with handwritten English texts from different writers, which can be used to train and test text recognition, writer identification, text segmentation, etc. All forms, text lines, words and sentences extracted are available for downloading. A XML with the meta-information of the text lines is also available. The XML information contains the description of all words in the text line. The coordinates of all the text components are also described.

The ANN segmentation method described in this paper was evaluated using all the handwritten text line from the writers of a subset denoted by C03 in the IAM database. We have ignored handwritten lines with XML information that indicates a segmentation error. Thus, 489 image lines were used to build the data sets for training and testing.

We have used the handwritten text line of each writer separately for training and testing (user-dependent evaluation). Two handwritten text lines were used for training, another two for validation and the remaining lines were used for testing. In this kind of experiment, one can achieve better rates for similar user writing styles.

The experiments performed here considered a margin of error (explained at the end of Section II-A.2) of three pixels. Figure 5 shows an example of the distance between the margin of error and the Bounding Box of the word.

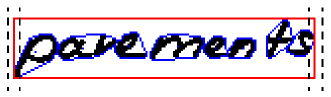

Fig. 5. The dotted line represents the margin of error adopted by the automatic evaluation procedure and the rectangle represents the XML bounding box.

In our experiments, two parameters were empirically defined to achieve the best segmentation error rate using the ANN method presented in this paper:

- Number of neurons in the hidden layer: in the range tested $[5, 50]$, the number of neurons in the hidden layer that produced the best performance was 30.
- Input size: the amount of patterns ("Pattern Grouping" size) used as input for the ANN that produced the best result was 40. The range tested was $[5, 50]$.

A. Post-processing

In order to improve the segmentation performance, we have developed a post-processing technique, which consists of using a sliding window over the sequence of classified patterns to change the pattern classification. If the patterns located in the window neighborhood have the same classification, then we change the window patterns to the same class as the neighbor patterns. Otherwise, no changes are performed. The size of the window must be empirically defined.

Figure 6 illustrated the over-segmentation and under-segmentation error rates using our post-processing technique. The horizontal axis represents the sliding window size and the vertical axis represents the error rate produced by over-segmentation and under-segmentation. Note that the under-segmentation error rate increases with the window enlargement. This occurs because the post-processing technique forces a larger sequence of patterns to be classified as a unique word or space between words. The opposite behavior is observed in the over-segmentation error rate.

According to Figure 6, the system can be adjusted to increase the over-segmentation error rather than the under-segmentation error, or vice-versa. This can be useful for adjusting the system to different styles of writing. Using a Size 4 Window, the Equal Error Rate is achieved (over-segmentation and under-segmentation error rate $\approx 4\%$). The error rates in Figure 6 were achieved from the mean of error of all handwritten text lines tested.

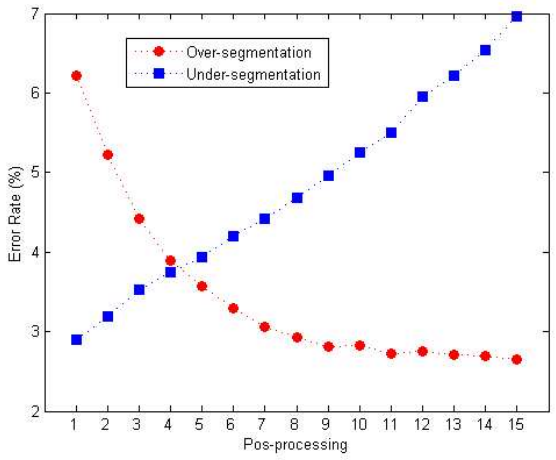

Fig. 6. Over and under-segmentation error rate.

B. System evaluation

For a better evaluation of the ANN segmentation method, the Convex Hull segmentation method described in [3] was developed. The accuracy of both methods was evaluated using the same data set and the same error margin of three pixels was considered. Table II displays the error rate achieved by the different methods.

- Convex Hull technique with the best configuration.
- ANN without post-processing (Window 0).
- ANN with best post-processing performance. The was achieved with the Size 9 Window (Window 9).

The ANN error rates were obtained from the average of 10 runs.

TABLE II
ERROR RATES OF CONVEX HULL AND ANN BASED METHOD WITH AND WITHOUT POST-PROCESSING.

		Window 0			Window 9		
ID	CH	Over	Under	Total	Over	Under	Total
150	10.34	11.73	3.16	14,89	3.30	4.53	***7.83***
151	18.08	7.33	6.16	***13.49***	2.80	16.19	18.99
152	***2.66***	7.93	1.43	9.36	2.54	2.03	4.56
153	***3.62***	5.54	1.87	7.40	3.51	3.04	6.55
154	4.53	7.87	0.54	8.41	1.54	2.02	***3.55***
155	24.44	11.33	0.29	11.62	3.16	1.93	***5.08***
$\overline{x}$	10.61	8.62	2.24	10.86	2.80	4.95	***7.76***

Figure 7 presents six box-plots of the post-processing accuracy. Nearly all the box-plots suggest that an optimum post-processing window size can be obtained for each writer. For example, a Size 9 Window is the best choice for post-processing for User 154, achieving 96.45% accuracy. The same behavior did not occur in the User 151 box-plot, as the standard deviations for this writer‘s error rate were the largest.

IV. CONCLUSIONS

The present paper addressed the problem of sentence segmentation. Our approach seeks to overcome inherent difficulties in the Gap Metrics approach, such as the heuristics needed to optimize and adapt the system to different applications in handwritten sentence segmentation; and the vocabulary limitation in other segmentation methods.

We presented an ANN-Based approach for off-line handwritten sentence segmentation. Assessments were performed under writer-dependent conditions on a sub-set from the IAM Database. Our experiments demonstrated that the ANN-based approach achieved better results for more writers in comparison to the Convex Hull segmentation method. No heuristics were used to adapt or improve system performance. Our method is learning-based and is therefore more appropriate for use in segmentation tasks. In future work, the proposed method should be tested under writer-independent conditions.

REFERENCES

[1] G. Seni and E. Cohen, ”External word segmentation of off-line handwritten text lines,” *Pattern Recognition*, vol. 27, pp. 41-52, 1994.

[2] U. Mahadevan and R. C. Nagabushnam, ”Gap metrics for word separation in handwritten lines,” *Third International Conference on Document Analysis and Recognition*, vol.1, pp. 124-127, 1995.

[3] U.V. Marti and H. Bunke, ”Text Line Segmentation and Word Recognition in a System for General Writer Independent Handwriting Recognition,” *Proc. Sixth Intl Conf. Document Analysis and Recognition*, pp. 159-163, 2001.

[4] Manmatha, R., Rothfeder, J.L., ”A Scale Space Approach for Automatically Segmenting Words from Historical Handwritten Documents”, *IEEE Transactions on Pattern Analysis And Machine Intelligence*, vol. 27, pp. 1212-1225, 2005.

[5] M. Morita, R. Sabourin, F. Bortolozzi and C. Y. Suen, ”Segmentation and recognition of handwritten dates: an HMM-MLP hybrid approach”, *International Journal on Document Analysis and Recognition*, pp. 248-262, 2004.

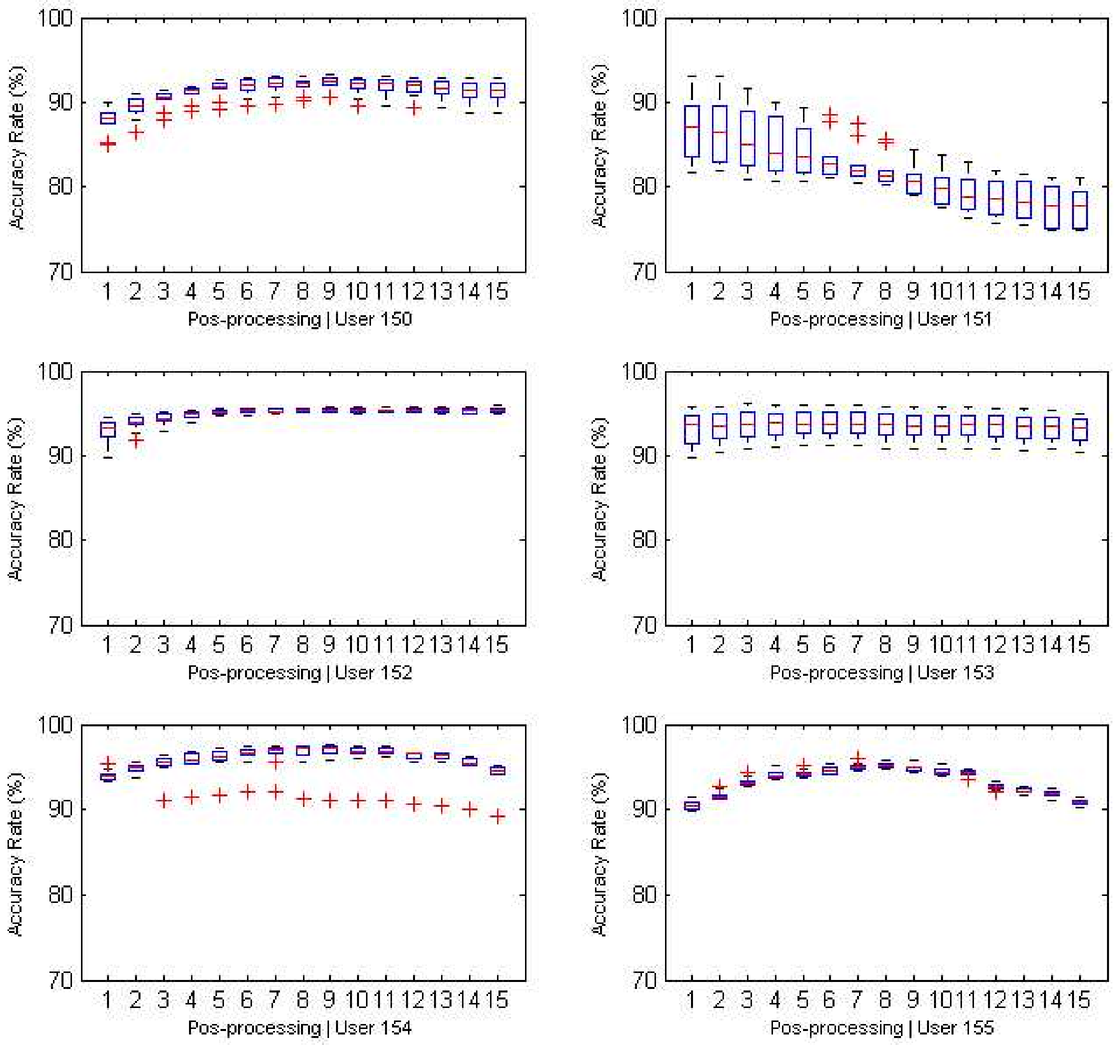

Fig. 7. Box-Plot of the Post-Processing Accuracy Rates

[6] F. Lthy, T. Varga and H. Bunke, ”Using Hidden Markov Models as a Tool for Handwritten Text Line Segmentation”, *Ninth International Conference on Document Analysis and Recognition*, vol.1, pp. 8-12, 2007.

[7] U.V. Marti and H. Bunke. ”Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system”. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 15(1): 65-90.2001.

[8] IAM Handwriting Database 3.0. Available in: {http://www.iam.unibe.ch/ fki/iamDB/}