

# Word Extraction from On-Line Handwritten Text Lines

Marcus Liwicki, Mathias Scherz and Horst Bunke  
Institute of Computer Science and Applied Mathematics  
University of Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland  
{liwicki, scherz, bunke}@iam.unibe.ch

## Abstract

*In this paper we investigate the word extraction task in on-line recognition of cursive handwritten text lines. For the segmentation we propose a method which is based on the assumption that the size of gaps between consecutive words may considerably vary, but humans usually leave more whitespace between two consecutive words than between two connected components that belong to the same word. We use several metrics known from off-line word segmentation for measuring the distances between two adjacent components. Then we apply different procedures to get the initial threshold for segmentation. Using these techniques we could significantly increase the segmentation rate compared to methods which are usually applied in on-line text recognition systems.*

## 1. Introduction

Word extraction is an important preprocessing step in most handwritten text recognition systems, because it allows one to reduce the difficult problem of recognizing a complete line of text to the simpler problem of recognizing single words [1, 12]. In on-line recognition systems the segmentation is usually based on the horizontal extend of the pen-movements to the right when the pen is lifted up. If this length exceeds a certain threshold it is assumed that a new word starts. In off-line recognition systems more refined procedures are typically used. The determination of a threshold for distinguishing between intra-word gaps and inter-word gaps has been investigated in [4] as well as methods for clustering the gaps between the connected components [5, 6, 9].

There exists some work in the field of on-line segmentation, but it is mainly based on text/graphic segmentation or structure analysis [3, 15]. In [11] the segmentation task is considered with more attention. It is based on the horizontal distance information as in this paper. The system in [11] uses neural networks for classification while the sys-

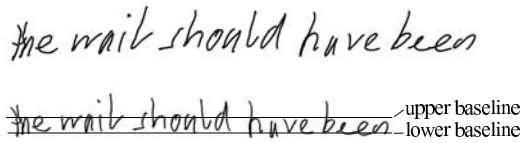
tem in this paper is based on a recently proposed assumption in off-line word segmentation [14]. The method in [14] is based on the assumption that the sizes of gaps between consecutive words may considerably vary, but humans usually leave more whitespace between two consecutive words (inter-word gaps) than between two connected components that belong to the same word (intra-word-gaps) up to some predefined factor. In this paper we propose a method for on-line word segmentation, which is based on the same assumption. The method first segments a line of text into sequences of connected components where the gaps between two sequences are larger than a previously calculated threshold. Then it further splits all sequences whenever the largest gap inside a sequence is larger than a fraction of the gaps that separate the considered sequence from other sequences.

For the evaluation of word segmentation algorithms there exist several measures [5, 9]. These measures take different aspects into account as the task of word segmentation can be interpreted in various ways. It can be interpreted as a word extraction task [14], a gap classification task [9], or the task of finding inter-word gaps. We present our results using three different measures and show that we can achieve improvements with all of them.

The rest of this paper is organized as follows. Section 2 presents the steps which are performed before the segmentation method is applied. In Section 3 different procedures for computing the initial threshold are described. Section 4 presents the proposed segmentation method. The results of our experiments are given in Section 5 and finally, Section 6 concludes the paper and proposes future work.

## 2. Preprocessing and gap width estimation

First, some preprocessing steps are applied to the handwritten text lines for the purpose of normalization (see Fig. 1). In our system we apply standard preprocessing operations that have been used in other handwriting recognition systems before [2, 7, 13]. As a first step the text lines are normalized with respect to slant and skew. After that we



**Figure 1. Example text line before and after normalization**

assume that words can be separated from each other by vertical cuts<sup>1</sup>. Secondly, the upper and lower baseline are estimated for each text line, which allows us to perform height normalization. Another preprocessing step is width normalization, where the number of characters is estimated using the number of white runs<sup>2</sup> between the upper and the lower baseline. Finally, punctuation marks are removed from the input data using some simple heuristics. This step is useful because punctuations often cause errors during the segmentation of a text line [5, 14]. An example of a text line before and after normalization is given in Fig. 1.

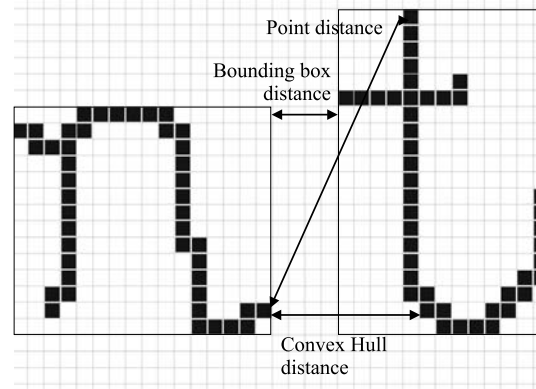
A connected component, or component for short, consists of one or several horizontally overlapping strokes. In the procedure proposed in this paper the distances between adjacent components have to be calculated. For these distances there exist several gap metrics (see Fig. 2):

- *Bounding box distance*: the width of the white gap in the vertical projection of the components.
- *Convex hull distance*: the minimal white run-length between the convex hulls of both components. If the components do not overlap vertically, the bounding box distance is taken.
- *Point distance*: the distance between the last point of the first component and the first point of the second component. This distance can be modified by using the mean of the last/first  $n$  points.

In addition to geometric information, it could be helpful to use the time information of the gaps to define a gap metric. We have investigated two possibilities to include time information. First we tried to use only the time distance. Second we scaled the time distance with a factor and added it to the geometric distance. This factor was optimized on the validation set.

<sup>1</sup>This assumption is satisfied by more than 99% of the data

<sup>2</sup>A white run is a series of white pixels connecting two black pixels in one pixel row.



**Figure 2. Illustration of the distance measures on a simplified example**

### 3. Initial threshold computation

In this paper we investigated three methods for calculating the initial threshold that is used to split a line of text into sequences of connected components. First we used a fixed value  $f$  as it is done in most handwriting recognition systems. This is reasonable since width normalization takes place during the preprocessing.

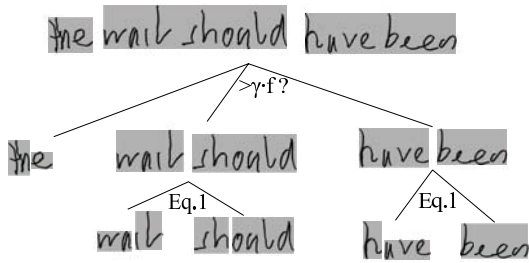
As an alternative we took the thresholds proposed in refs. [10, 14] for off-line representation of the handwriting. Both thresholds are based on the white run-lengths between the upper and lower baseline of the handwriting. These lines separate the ascenders and the descenders from the middle part of the handwriting. The thresholds are based on the following statistics  $f$ :

- *Median white run-length (MWR)*: the median of the set of white run-lengths between the two lines.
- *Average white run-length (AWR)*: the median of the number of white pixels in a row divided by the median number of black-white transitions in a row.

In the basic version of our segmentation method (see below) the text line is divided at all gaps which are larger than the threshold  $\gamma * f$ , where  $\gamma > 0$  is a parameter optimized on a separate validation set in our experiments and  $f$  is the computed statistic.

### 4. Proposed method

Our word segmentation algorithm is based on the assumption that for all words  $w_i$  the gaps within  $w_i$  are smaller than the gaps between  $w_i$  and its left and right neighbor  $w_{i-1}$  and  $w_{i+1}$ . In our algorithm, we first divide



**Figure 3. Proposed method for an example text line**

the text line at all gaps that are larger than the initial threshold. Then all remaining sequences of consecutive connected components are taken into account. For each remaining sequence  $S$  the maximal gap  $maxgap$  inside the sequence and the gaps  $leftgap$  and  $rightgap$  to the left and right hand side of  $S$  are considered. If they fulfill the condition

$$\alpha * maxgap \geq \min\{leftgap, rightgap\} \quad (1)$$

then the sequence is further divided into two subsequences, where  $\alpha$  is a predefined parameter. This is repeated recursively until no more sequences fulfill the condition in Eq. 1. Note that  $\alpha$  is optimized on a separate validation set in our experiments.

Figure 3 illustrates how the algorithm works. First all gaps which are larger than the initial threshold are considered as segmentation points. This results in the three sequences of connected components “the”, “work should”, and “have been”. Then the largest gap of each sequence is tested if it fulfills the condition of Eq. 1. As a result the sequences “work should” and “have been” are further divided. The algorithm stops at this point, because the remaining gaps are too small. Note that the blanks denote the gaps which are considered in the corresponding next step.

## 5. Experiments and results

All experiments have been conducted on a subset of the the IAM-OnDB [8], which is a large database of on-line handwritten text lines. The selected subset consists of 100 handwritten texts with about eight lines each. For optimizing the parameters  $\gamma$  and  $\alpha$  of Sections 3 and 4 we used an additional validation set of 20 writers. Each text was produced by a different writer. Hence a total of 120 writers were involved.

For measuring the performance of a segmentation algorithm there exist several methods. In this paper we used three different performance measures:

- *Word extraction rate (WER)*: with this measure the segmentation task is considered as a word extraction task [5, 14]

$$WER = \frac{\text{number of correctly extracted words}}{\text{number of all words}} \quad (2)$$

- *Gap classification rate (GCR)*: the segmentation task is considered as a gap classification task which classifies the gaps as inter-word gaps or intra-word gaps [9]

$$GCR = \frac{\text{number of correctly labeled gaps}}{\text{number of all gaps}} \quad (3)$$

- *Gap accuracy (GA)*: the segmentation task is considered as an inter-word gap finding task

$$GA = \frac{\text{correctly found gaps} - \text{gaps wrongly found}}{\text{all gaps}} \quad (4)$$

Depending on the underlying performance measure, the results vary. Usually, the best results are reported with the second measure, because it is rather simple to classify most short gaps as intra-word gaps.

First, we investigated the gap metrics presented in Section 2. We found out that the bounding box distance was the best metric on the validation set. Integrating on-line information, i.e. temporal information, into the distance measure did not increase the performance. A possible reason for this is that, on the one hand, the writers sometimes made a break within a word for moving to the right in front of the whiteboard. On the other hand, sometimes they quickly wrote a couple of words without pausing. In this respect the whiteboard handwriting data may be different from handwriting produced on an electronic tablet.

Table 1 shows the results of all segmentation methods using different thresholds and accuracy measures. The baseline method only uses the initial threshold for segmentation. For initial threshold computation the average white run-length turns out to be the best choice (column ‘basic’ in Table 1). However, using a fixed threshold produces good results as well. This is because the text lines have been width normalized before segmentation.

In Table 1 it can be observed that the new method always outperforms the baseline system. Bold entries denote that the increase compared to the corresponding baseline system is statistically significant on the 95% level. This confirms our statement that having a closer look at the segmentation task improves the word segmentation rate.

The differences of the three performance measures can also be seen in the results. The highest values are obtained using the gap classification measure, while the word extraction measure and the gap accuracy are nearly on the same level. A possible reason for the high accuracy of the gap

**Table 1. Performance of the segmentation methods for the three measures of Eqs. 2-4 on the test set**

method		basic	new	error red.
measure	threshold			
WER	FIX	85.36	<b>86.42</b>	7.24
	MWR	85.32	<b>86.08</b>	5.18
	AWR	85.95	<b>86.58</b>	4.48
GCR	FIX	95.64	95.69	1.15
	MWR	95.54	95.66	2.69
	AWR	95.69	95.75	1.39
GA	FIX	87.22	<b>87.50</b>	2.19
	MWR	86.95	<b>87.55</b>	4.60
	AWR	87.41	<b>87.73</b>	2.54

classification measure is that there are many intra-word-gaps which are only small. For those gaps the classification is rather simple.

## 6. Conclusions and future work

In this paper we presented a word extraction system for on-line data. In the literature only little attention has been paid to this specific task of on-line handwriting recognition. We applied state-of-the-art methods for initial threshold computation and for hierarchically segmenting a text line. Our results show that there is a significant potential in increasing the segmentation performance over simple heuristic methods. We could increase the word extraction rate from 85.36% in the baseline method where a fixed threshold is used to 85.95% by computing the initial threshold with the AWR method. By applying the new word segmentation algorithm the word extraction rate could be further increased to 86.58%, which is statistically significant.

We also showed that the definition of the measure has a great impact on the level of the reported results. While the word extraction rate is about 86%, the gap classification rate is already over 95%. This should be considered when results of different papers in literature are compared with each other.

In future we plan to combine several word segmentation systems to a multiple classifier system. This is reasonable since the different approaches produce diverse results. Also the values for  $\alpha$  and  $\gamma$  can control oversegmentation and undersegmentation. Thus a combination of different segmentation systems based on varying these two parameters might also improve the performance.

Initial experiments of a text line recognizer have shown

a slightly improvement of the word recognition rate when the segmentation information is used. Before that, the word segmentation was integrated in the recognition process, i.e. the character *space* was also modeled by the recognition system. We plan to further extend our set of recognition experiments to find the best way of integrating the word segmentation information into the recognition system.

## Acknowledgments

We would like to thank Tamas Varga for his valuable hints regarding to the segmentation methods.

## References

- [1] E. Caillault, C. Viard-Gaudin, and P. Lallican. Training of hybrid ann/hmm systems for on-line handwritten word recognition. In *Proc. 12th Conf. of the International Graphonomics Society*, volume 1, pages 212–216, 2005.
- [2] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: the NPen++ recognizer. *Int. Journal on Document Analysis and Recognition*, 3(3):169–180, 2001.
- [3] A. K. Jain, A. M. Namboodiri, and J. Subrahmonia. Structure in on-line documents. In *Proc. 6th Int. Conference on Document Analysis and Recognition*, pages 844–848, 2001.
- [4] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis. An unconstrained handwriting recognition system. *International Journal on Document Analysis and Recognition*, 4(4):226–242, 2002.
- [5] S.-H. Kim, C. B. Jeong, H. K. Kwag, and C. Y. Suen. Word segmentation of printed text lines based on gap clustering and special symbol detection. In *Proc. 16th Int. Conf. on Pattern Recognition*, volume 2, pages 320–323, 2002.
- [6] S.-H. Kim, S. Jeong, G.-S. Lee, and C. Y. Suen. Word segmentation in handwritten Korean text lines based on gap clustering techniques. In *Proc. 6th Int. Conf. on Document Analysis and Recognition*, pages 189–193, 2001.
- [7] M. Liwicki and H. Bunke. Handwriting recognition of whiteboard notes. In *Proc. 12th Conf. of the International Graphonomics Society*, pages 118–122, 2005.
- [8] M. Liwicki and H. Bunke. IAM-OnDB - an on-line English sentence database acquired from handwritten text on a whiteboard. In *Proc. 8th Int. Conf. on Document Analysis and Recognition*, volume 2, pages 956–961, 2005.
- [9] U. Mahadevan and S. N. Srihari. Hypothesis generation for word separation in handwritten lines. In *Proc. 5th Int. Workshop on Frontiers in Handwriting Recognition*, pages 453–456, 1996.
- [10] U.-V. Marti and H. Bunke. Text line segmentation and word recognition in a system for general writer independent handwriting recognition. In *Proc. 6th Int. Conference on Document Analysis and Recognition*, pages 159–163, 2001.
- [11] L. Oudot, L. Prevost, and M. Milgram. An activation-verification model for on-line texts recognition. In *Proc. 9th Int. Workshop on Frontiers in Handwriting Recognition*, pages 485–490, 2004.

- [12] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):63–84, 2000.
- [13] M. Schenkel, I. Guyon, and D. Henderson. On-line cursive script recognition using time delay neural networks and hidden Markov models. *Machine Vision and Applications*, 8:215–223, 1995.
- [14] T. Varga and H. Bunke. Tree structure for word extraction from handwritten text lines. In *Proc. 8th Intl. Conf. on Document Analysis and Recognition*, volume 1, pages 352–356, 2005.
- [15] M. Ye, H. Sutanto, S. Raghupathy, C. Li, and M. Shilman. Grouping text lines in freeform handwritten notes. In *Proc. 8th Int. Conference on Document Analysis and Recognition*, pages 367–373, 2005.