Handwritten Character Segmentation Using Transformation-Based Learning

E. Kavallieratou, E. Stamatatos, N. Fakotakis, and G. Kokkinakis
Dept. of Electrical & Computer Engineering
University of Patras, 26500 Patras, Greece
Tel. ++30-61-991722, fax ++30-61-991855
ergina@wcl.ee.upatras.gr

Abstract

This paper presents a character segmentation algorithm for unconstrained cursive handwritten text. The transformation-based learning method and a simplified variation of it are used in order to extract automatically rules that detect the segment boundaries. Comparative experimental results are given for a collection of multi-writer handwritten words. The achieved accuracy in detecting segment boundaries exceeds 82%. Moreover, limited training data can provide very satisfactory results.

1. Introduction

The recognition of cursive handwritten text is one of the hardest cases in Optical Character Recognition (OCR). Nevertheless, it is very popular as research subject due to its numerous potential applications. The hitherto proposed methods can be distinguished in the holistic approaches, where character segmentation is not performed, and the segmentation based approaches. According to Casey and Lecolinet [1]:

> *"Character segmentation is an operation that seeks to decompose an image of a sequence of characters into subimages of individual symbol."*

Several surveys of the research in character segmentation have been published [1-3]. Lu and Shridar [2] claim that:

> *"Recognition based on the entire word is difficult and complicated. It is also time and memory intensive since all the representations of the words in the vocabulary under investigation need to be kept and matched with the input word image. Another drawback is that incorrectly spelled words cannot be recognized since they do not belong to the vocabulary."*

The majority of the existing segmentation algorithms use rules that have been extracted by observing empirically the human writing. Thus, although sometimes the results may be satisfying, there is no guarantee that all possible rules or the optimal rule set have been found. On the other hand, some methods that have been used in order to extract the required rules automatically (e.g., hidden Markov models, neural networks [4] etc.) provide rules that are based on numerical data. As a consequence, the understanding and the improvement of the extracted rules is very difficult.

In this paper we present a character segmentation algorithm that is based on a simplified version [5] of the transformation-based learning theory [6], a technique that has been widely applied to natural language processing. This technique extracts automatically the required knowledge in the form of IF-THEN rules. Moreover, the rules are easily comprehensible, since they are not based on opaque tables of statistics.

An overview of the proposed methodology is shown in figure 1. In this paper we assume that word segmentation has already been performed and the words have been corrected in terms of shift and slant by applying the methods described in [7]. The character segmentation procedure includes two stages: A presegmentation stage provides a first estimation of the segment boundaries followed by a machine learning algorithm that refines the presegmentation and provides the final segment boundaries.

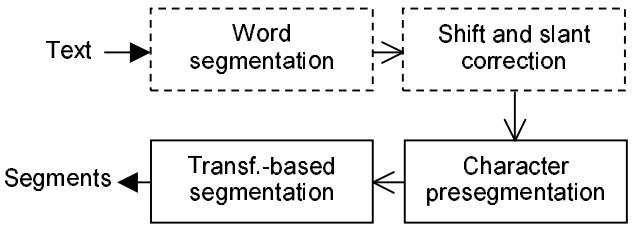

Figure 1. The proposed methodology.

In the next section we briefly describe the presegmentation stage. The segmentation is described in section 3. Finally, some experimental results are given in section 4 and some conclusions are drawn in section 5.

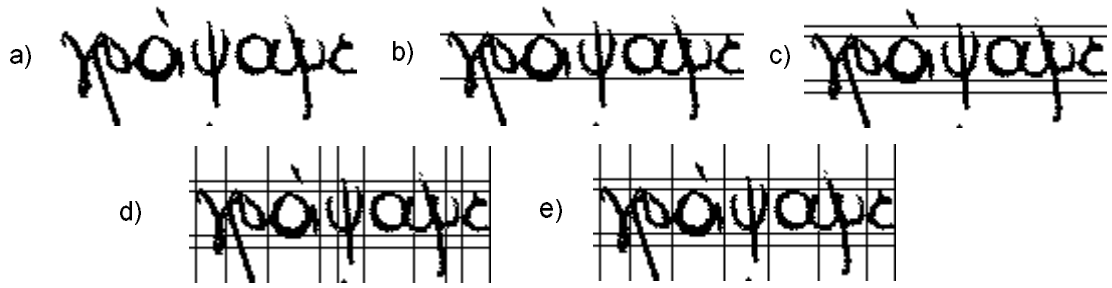

Figure 2. (a) the sample word, (b) the main body determined by our algorithm, (c) the main body increased ¼ towards up and down, (d) the word after the presegmentation stage, and (e) the final segmented word.

2. Presegmentation

During this stage the word is processed and a first estimation of the position of the possible segment boundaries is carried out. Both ascenders and descenders of a word complicate the tracing of possible segment boundaries. In order to avoid this problem, our approach focuses on the main part of the word. The main part is determined by using the horizontal histogram of the word and excluding the upper and lower parts where the value of the histogram falls under the 1/3 of its maximum. The success rate of this technique in our experiments was 97%, presenting a drawback only in short words (i.e., words with less than three characters). Nevertheless, we increased this zone ¼ of its height towards up and down since in handwritten words the characters are not always written horizontally.

Characters without ascenders and descenders, like a, e, c, etc., present almost the same height and width even when they are handwritten, although this varies from person to person. The same happens with the rest of the characters, if we exclude their ascenders and descenders. Then, the only exceptions in the English alphabet are the characters *w, m, i, j, l* and *ω, ι* of the Greek alphabet. However, even in these cases, if a general estimation is required, the width can be expressed as (3/2)*height (*m, w, ω*) or 1/2*height (i, *j, l, ι*). Since in the presegmentation stage only an initial estimation of the possible character segments is performed, the main height of the main part of the word (which from now on will be referred to as *main height*) is taken into account.

After the calculation and the smoothing of the vertical histogram of the word, the minima of the histogram are extracted. The amount of considered minima is limited to the quotient of the length of the word, to the half of the main height. This is the number of minima when all the characters of the word are *i, l, j*. Moreover, the distance between these minima should be at least the 1/3 of the height, since a character is unlikely to fit in less than this width. The segment boundaries that intersect (not touch) twice a character are excluded since they are extremely likely to be false.

If the distance between two successive segment boundaries is longer than the main height, then we look for an intermediate segment boundary by finding the minimum of the histogram in this area.

The final step of the presegmentation is the vectorization of the remaining segments. Here, it is crucial to keep the information that best distinguishes the segments. Thus, the selected 7-parameter vectors consist of the following information:

- The position that the starting segment boundary intersects a character (if it does), with respect to the main height.
- The value of the vertical histogram at the starting segment boundary of the segment.
- The width of the current segment as a proportion of the main height.
- The position of the two longest horizontal (or almost horizontal) strokes (if they exist) in the segment, with respect to the main height.
- The position of the two longest vertical (or almost vertical) strokes (if they exist) in the segment, with respect to the segment width.

3. Transformation-Based Segmentation

Transformation-Based Learning (TBL) is a machine learning theory that requires already annotated data in order to extract the appropriate knowledge in the form of simple rules. In particular, the rules are of the following format:

IF *triggering environment*
THEN *transformation*

where the *transformation* changes the state of a tag (e.g., a segment boundary) if the condition described in the *triggering environment* is valid. Initially, the training data are annotated based on an initial-state annotator. The extraction of rules is performed via an iterative procedure. During each iteration, all the possible transformations are tested and the one that achieves the best results (by comparing the derived data with the already manually annotated data) is selected.

Table 1. Results of the presegmentation procedure.

Segment boundaries	Training data	Test data	Total data
True	2,096	898	2,994
False	1,102	491	1,593
Total	3,198	1,389	4,587
Accuracy (%)	65.5	64.7	65.3

Table 2. The performance of the machine learning techniques.

Learning theory	Baseline (%)	Accuracy (%)	True segment boundaries	False segment boundaries
Traditional TBL	64.7	77.8	1,081	308
Simplified variation	64.7	82.4	1,145	244

An ordered list of rules is, thus, learned. Learning stops when no rule manages to improve the accuracy of the annotated corpus beyond a predefined threshold.

TBL is an application-independent theory and has been applied successfully to a wide range of natural language processing tasks, including part-of-speech tagging [6], text chunking [8], and dialog act tagging [9]. In order to be adapted to a specific application the following have to be defined:

- The initial state annotator
- The space of allowable transformations (rules and the triggering environments)
- The objective function for comparing the corpus to the truth and choosing a transformation

TBL does not take into account the features of a certain application. This fact may cause problems in the accuracy of the model as well as considerable losses regarding the training time cost. In order to overcome these problems a variation of the traditional TBL has been proposed [8]. This variation suits applications that are characterized by a limited number of possible transformations. Notice that character segmentation is one problem of that category since a tag may take only two values: 0 representing a false segment boundary and 1 representing a true segment boundary. Therefore, the possible transformations are only two (i.e., $0 \rightarrow 1$ and $1 \rightarrow 0$).

The above variation of TBL applies first all the rules that perform the least likely transformation regardless the errors they may produce. Afterwards, all the rules that perform the next likely transformation are applied. Moreover, all the rules are extracted by comparing only one time the training data to the already annotated data and, therefore, the required training time cost is considerably lower. Considering only two possible transformations, the annotation procedure of an unseen case is performed by applying first the less-likely transformation rules (i.e., $1 \rightarrow 0$) and, then, applying the rules that perform the most-likely transformation (i.e., $0 \rightarrow 1$). In figure 2, an example of the different stages of our algorithm is shown.

4. Performance

The proposed methodology has been applied to a collection of words consisting of approximately 500 handwritten English and Modern Greek words taken from approximately 250 different writers. These words were divided into training and test data in a 2:1 ratio. The presegmentation module was used to provide an initial estimation of the character segment boundaries for both training and test data. The presegmentation algorithm found all the actual segment boundaries plus some false ones. The results of this procedure are shown in table 1. These data have been annotated by tagging the true and false segment boundaries manually.

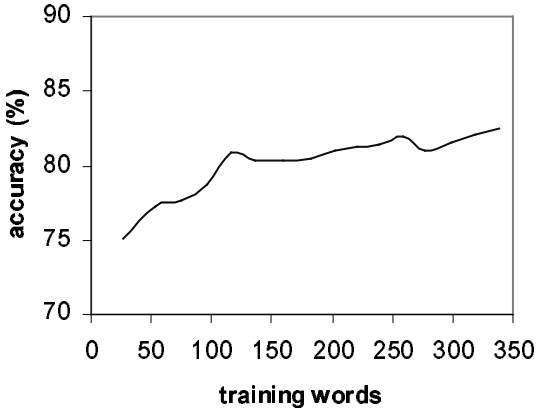

Figure 3. Training data vs. accuracy.

Both the traditional TBL theory and its simplified variation have been applied to the training data in order to extract the disambiguation rules automatically. The triggering environment of a segment boundary consists of the 7-parameter vector of the segment that starts with this boundary (current segment) plus the width of the segments that immediately precede and follow the current segment, as a proportion of the main height. As regards the traditional TBL, the current values of the segment

boundaries (i.e., true or false) that immediately precede and follow the segment boundary in question are also taken into account. Thus, the triggering environment consists of totally 11 and 9 variables for TBL and its simplified variation respectively. Notice that in both cases the values of the vectors were standardized.

Comparative results of the application of the extracted rules to the test data are given in table 2. The baseline column refers to the performance of the presegmentation module. The performance of the simplified variation is considerably higher than that of the traditional TBL algorithm. The relationship of training data with the disambiguation accuracy is shown in figure 3. Notice that the test data were always the same (see table 1). It is clear that approximately 120 training words provide very good accuracy results.

5. Conclusion

In this paper we presented an approach to handwritten character segmentation, a task that is crucial for enhancing the results of a handwritten character recognition application. The segmentation procedure has two phases. Initially, a first estimation of the segment boundaries is performed based on a simple algorithm. The theory of transformation-based learning is, then, used in order to decide, on the basis of disambiguation rules extracted in a training phase, which segment boundaries, proposed by the presegmentation module, are actual segment boundaries.

The presented methodology has been tested on a multi-writer collection of handwritten words and the accuracy exceeded 82%. Notice that this is not an upper bound for the recognition module. A great part of the falsely detected boundaries (about 11% of the total segments) is caused by the connection of cursive characters. Such errors may cause the insertion of redundant, but easily detectable, characters during the recognition stage. In addition, a robust recognition algorithm has to be able to recognize characters that have not been segmented perfectly. The comparison of the achieved accuracy to other systems is not easy since, in the literature, the performance of a segmentation algorithm is usually combined with the performance of the recognition module.

Our approach is language-independent and has been tested on bilingual data. The reported results can, however, be considerably improved by training the system for a specific writer. Since the extracted rules are based on vectors consisting of human comprehensible information rather than opaque numerical data, a further improvement of the reported accuracy by refining the automatically extracted rules manually is possible.

The proposed approach requires training. This may cause some problems since character segmentation is just a pre-processing task for character recognition. However, by using the same samples for training both the segmentation and the recognition modules, the training requirements of the overall system are minimized. Moreover, as shown in figure 3, we can achieve very satisfactory results by using relatively limited training data.

References

[1] Casey, R. and E. Lecolinet, "A Survey of Methods and Strategies in Character Segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7), pp. 690-706, 1996.

[2] Lu, Y. and M. Shridar, "Character Segmentation in Handwritten Words - An Overview", *Pattern Recognition*, 29(1), pp. 77-96, 1996.

[3] Elliman, D. and I. Lancaster, "A Review of Segmentation and Contextual Analysis Techniques for Text Recognition", *Pattern Recognition*, 23(4/5), pp. 337-346, 1990.

[4] Giloux, M., "Hidden Markov Models in Handwriting Recognition", *Fundamentals in Handwriting Recognition*, S. Impedovo ed., NATO ASI Series F: Computer and System Sciences, 24, Spinger Verlang, 1994.

[5] Stamatatos, E., N. Fakotakis, and G. Kokkinakis, "Automatic Extraction of Rules for Sentence Boundary Disambiguation", In *Proc. of the Workshop in Machine Learning in Human Language Technology, Advance Course on Artificial Intelligence,* pp. 88-92, 1999.

[6] Brill, E., "Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging", *Computational Linguistics*, 21(4), pp. 543-565, 1995.

[7] Kavallieratou, E., N. Fakotakis, and G. Kokkinakis, "New Algorithms for Skewing Correction and Slant Removal on Word-Level", In Proc. of *ICECS'99*, v.2, pp. 1159-1162, 1999.

[8] Ramshaw, L. and M. Marcus, "Text Chunking Using Transformation-based Learning", In *Proc. of ACL Third Workshop on Very Large Corpora.* pp. 82-94, 1995.

[9] Samuel, K., "Dialogue Act Tagging with Transformation-Based Learning", In *Proc. of the 17th International Conference on Computational Linguistics,* 1998.