Automatic Segmentation of the IAM Off-line Database for Handwritten English Text

Matthias Zimmermann, Horst Bunke
University of Bern
Institute of Informatics and Applied Mathematics
Neubrueckstrasse 10, CH-3012 Bern, Switzerland
{zimmerma,bunke}@iam.unibe.ch

Abstract

This paper presents an automatic segmentation scheme for cursive handwritten text lines using the transcriptions of the text lines and a hidden Markov model (HMM) based recognition system. The segmentation scheme has been developed and tested on the IAM database that contains off-line images of cursively handwritten English text. The original version of this database contains ground truth for complete lines of text only, but not for individual words. With the method described in this paper, the usability of the database is greatly improved because accurate bounding box information and ground truth for individual words (including punctuation characters) is now available as well. Applying the segmentation scheme on 417 pages of handwritten text a correct word segmentation rate of 98% has been achieved, producing correct bounding boxes for over 25'000 handwritten words.

1 Introduction

Large databases play a significant role in the development, evaluation, comparison and improvement of handwriting recognition systems. Since both the collection of the data and the production of the corresponding transcriptions (labeling) are expensive and time consuming tasks it is normally desired to reuse existing databases as far as possible.

Until recently the majority of publicly available databases for off-line handwriting recognition provide character and word images for limited tasks only. For example the NIST database [2] is based on a small vocabulary. In the database described in [15] only a single writer is represented. Other databases have been built for specific tasks like the recognition of handwritten legal amounts [7, 4] or postal addresses as in the CEDAR database [5, 3, 9]. More recently a database containing essays written by students was presented in [1].

Figure 1. Word segmentation result

In order to train and test systems which are able to recognize unconstrained English texts as described in [8] or [11, 12], the IAM database has been built at our institute. This database is available upon request and has been documented in [10, 13]. The original aim of this database was to provide training and test data for a segmentation-free recognition system for cursive handwriting. Consequently, ground truth in this database was acquired only on the level of complete text lines, but not on the level of individual words and punctuation characters.

Using the segmentation scheme described in this paper we can now provide accurate bounding box information for the individual words and punctuation characters for each form image in this database as well. Using this information as segmentation ground truth, large amounts of training and test data for a variety of handwriting recognition tasks in addition to the task originally considered become available. Examples of such tasks are the recognition of isolated handwritten words or the automatic measurement of the segmentation performance of handwritten text recognition systems. Fig. 1 gives an example of the result of the segmentation scheme presented in this paper.

This paper continues with the presentation of the proposed word segmentation procedure. Section 3 describes the assignment of word bounding boxes. Experiments and results are presented in Section 4 and conclusions are drawn in Section 5.

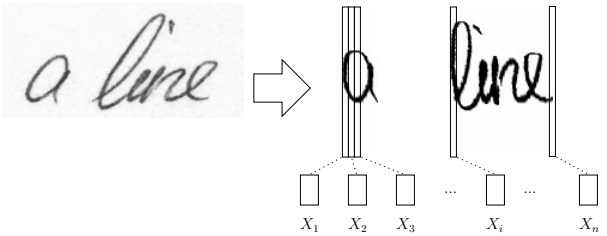

Figure 2. Text line normalization and extraction of a feature vector sequence

2 HMM based Word Segmentation

For each handwritten line of text in the IAM database, ground truth in terms of an ASCII transcription is available. The task considered in the following is to derive ground truth at the word level. This means that we need to assign to each word in the ASCII ground truth (including punctuation characters) the bounding box of the corresponding word in the text image. In order to automatize this task as far as possible we used the HMM based recognition system which has been developed previously at our institute to recognize lines of handwritten text [11]. Compared to this system the proposed segmentation system differs in the applied mode of the Viterby decoder, the word and the character modeling.

In contrast to the recognition of cursively handwritten text we do not try to recognize the individual words. Instead, we provide the transcription of a complete line of text to the Viterby decoder and use it in forced alignment mode. Using the Viterby decoder in forced alignment mode produces an optimal segmentation of the the text line into its words[1]. This procedure leads to very good segmentation results but requires well trained character HMMs.

In order to obtain reliable character HMMs and robust segmentation results all text line images have to be normalized to reduce the variability of the different writing styles. The normalization and feature extraction procedures used for the segmentation task considered in this paper are identical to the original recognition system [11].

Fig. 2 provides an example of the text line normalization and the extraction of the corresponding feature vector sequence $X = (X_1, X_2, \ldots X_i \ldots X_n)$. Using a sliding window technique a single feature vector X_i is extracted for each column of the normalized text line image. Consequently the horizontal coordinates of the image columns are directly represented by the indices of the corresponding feature vectors. This fact supports the accurate location of the word beginnings as we will see later.

[1]The segmentation produced by the Viterby decoder is optimal for a given HMM state sequence and a given feature vector sequence extracted from a text line.

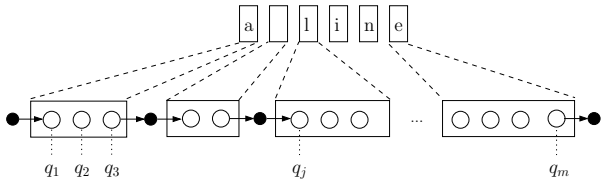

Figure 3. Generation of a state sequence by concatenation of character HMMs

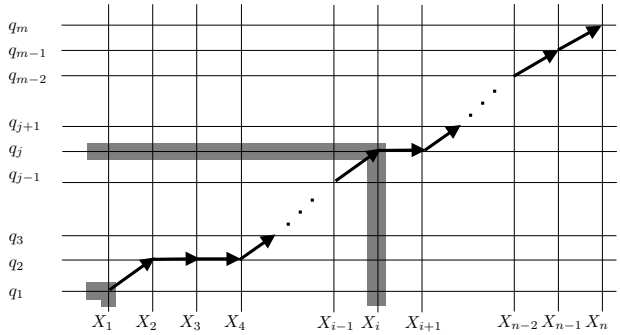

Figure 4. Forced alignment of a given feature vector sequence and a state sequence

The HMM state sequence $Q = (q_1, q_2, \ldots q_j \ldots q_m)$ used in the forced alignment mode can be generated by the concatenation of the trained character HMMs according to the transcription of the text line. This is shown in Fig. 3 where the character HMMs have been simplified by using fewer states and omitting the state transitions inside the models. All character HMMs in our system have a linear left to right topology which only allows transitions from a given state to itself and to its right neighbor.

Providing both the feature vector sequence X and the state sequence Q to the Viterby decoder the optimal alignment $\hat{A} = \arg\max_A P(X, A|Q)$ can be found by dynamic programming. $\hat{A} = ((X_1, q_1), \ldots (X_i, q_j) \ldots (X_n, q_m))$ represents therefore the most likely assignment of each X_k to a state q_l. The dynamic programming is illustrated in Fig. 4 using the example given in Fig. 2. $\hat{A}$ is represented as a sequence of arrows and the detected beginnings of the words 'a' and 'line' have been marked using gray shadows. Because each feature vector X_k has to be absorbed by exactly one state and each state q_l consumes at least one feature vector it follows that each A considered in the computation of $\hat{A}$ has to start with the assignment (X_1, q_1) and must end with (X_n, q_m). All elements of the path between these two positions have to respect the restrictions imposed by the topology of the character HMMs involved in the state sequence Q. If the length of the state sequence Q is greater

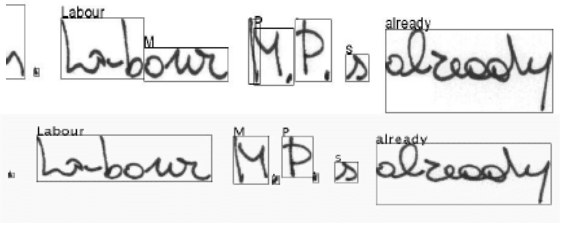

Figure 5. The effect of character depending HMM topologies

than the length of the feature vector sequence X no path A can be found and the forced alignment fails. In all other cases the forced alignment produces a path $\hat{A}$ and the beginning of a word in the normalized text line can be easily found, because the smallest index of all feature vectors assigned to the first state of a given word directly represents the horizontal coordinate for this word.

Unfortunately the coordinates of the word beginnings in the normalized text line can not directly by used to produce the word bounding boxes in the original image as shown in Fig. 1. This fact is caused by the normalization operations applied to the original image. A way to compute the bounding boxes for the original, non-normalized images using the word beginnings obtained by the scheme described above is provided in Sec. 3.

In order to allow the segmentation of all punctuation characters, both additional lexicon entries and corresponding hidden Markov models are needed. They are not supported by the original recognition system. In the original recognition system the hidden Markov models for all characters used a linear topology consisting of 14 states. The number of states has been determined by global optimization of the recognition rate for the complete text recognition system [11].

First segmentation experiments using forced alignment showed that the state sequence for parts of the text line could easily become longer than the width of the corresponding image parts[2]. In all those cases the Viterby decoder was forced to compensate the mismatch between the number of states in the model sequence and the number of available feature vectors by consumption of additional feature vectors from neighboring words. This normally resulted in segmentation errors.

Based on this observation the lengths of the space and punctuation models have been reduced to two states each. The effect of the character depending topology is shown in Fig. 5 where the first line has been segmented with HMMs of constant length 14 while the second line was segmented using the short models for the space and the punctuation models.

[2]Since punctuation characters have been treated as words and space models were inserted after every word, the state sequence for an abbreviation like 'M.P.' became (M, space, point, space, P, space, point, space) and contained therefore 112 states.

3 Assignment of Word Bounding Boxes

To find the word bounding boxes in the original text lines the algorithm described in this section uses the horizontal coordinates provided by the HMM based word segmentation procedure. As mentioned in the previous section, the resulting coordinates correspond to the beginnings of the words in the segmented text line. It is important to keep in mind that these positions represent the word boundaries for the normalized text line only. The mapping of these boundaries to the word bounding boxes in the original line images has been implemented using an intermediate representation of the text lines. For the assignment of connected components to words and the computation of the word bounding boxes the following algorithm has been used.

1. Binarize the original, non-normalized image.
2. Extract the connected components and filter out small components.
3. Label every remaining connected component by assigning a unique label value to all its pixels and record the coordinates of its bounding box in the original form image.
4. Apply the normalization scheme used in the HMM based segmentation procedure to the labeled connected components extracted in step 3.
5. Assign the normalized connected components to the word boundaries obtained from the HMMs according to their horizontal positions.
   - If a normalized component falls in between the left and the right boundary of a word, assign it to this word. Please note that the right boundary of a word is identical to the left boundary of the next word (see Fig. 6).
   - If a normalized component crosses a word boundary assign it to the word with the bigger overlap.
6. Use the bounding box coordinates recorded in step 5 of all connected components assigned to a given word to compute the corresponding word bounding box (defined as the minimal rectangle enclosing all bounding boxes of the connected components).

It could be verified by experiment that connected components belong to a single word or punctuation mark in most

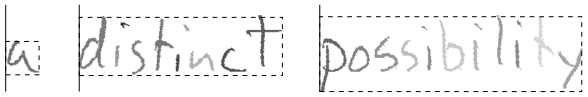

Figure 6. Assignment of normalized connected components to word boundaries

cases [3]. Based on this observation we defined word bounding boxes as the minimal rectangle including all connected components of a given word or punctuation character.

The assignment of the connected components as described in step 5 of the algorithm is visualized in Fig. 6. The long vertical lines indicate the word boundaries detected by the HMM based segmentation procedure and the dashed rectangles indicate the assignment of the connected components to the words. The normalized connected components are drawn using unique color values as described in step 3 of the algorithm.

4 Experimental Results

The proposed segmentation scheme has been applied on 417 forms, comprising 3703 text lines, from text categories A, B and C of the IAM database[4]. For the manual verification of the segmentation process, an image of each of the processed text lines is generated. The image shows the handwritten text, the corresponding transcription, the word bounding boxes, and the individual word labels (see Fig. 7 for an example).

The forms of both text categories A and B have been used during the development of the segmentation scheme. In several refinement steps the HMM based segmentation and the connected component assignment have been improved and transcriptions which did not match the handwritten text have been corrected. Fig. 7 shows an example of an incorrect transcription. The writer omitted the final period of the line. Because it is included in the lines transcription the component assignment step marks the letter 's' as the period.

In all experiments reported in this paper a word was considered to be segmented correctly when its bounding box was minimal, did not truncate any part of the word and the handwritten word image corresponded to the assigned word label. A text line or a complete form was considered to be correctly segmented when all its words were correctly segmented according to the definition give above.

[3]Using Otsu's [14] binarization algorithm only 33 exceptions in the 30'785 processed words have been detected. About 80% of the involved words did touch a word in the line above or below.

[4]The text categories are defined by the Lancaster - Oslo/Bergen corpus (LOB) [6]. Text category A stands for 'press: reportage', category B for 'press: editorial' and C for 'press: reviews'.

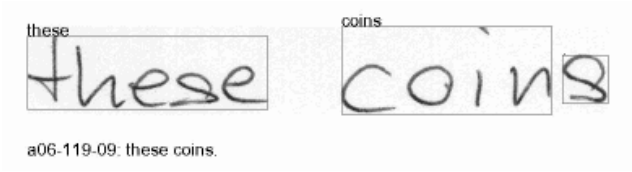

Figure 7. Interface for verification and manual correction of transcriptions

Set	Forms	Lines	Words
A	162 (62%)	1490 (93%)	11954 (98.3%)
B	116 (53%)	996 (87%)	8357 (98.1%)
C	139 (38%)	1217 (87%)	10474 (97.0%)

Table 1. Correct segmentation rates for forms, lines and words

The result of the proposed segmentation procedure is documented in Tab. 1. For each of the three considered text categories, the number of forms, text lines and words is given together with the correct segmentation rate. The handwritten texts from category C have been defined as test set. The correct word segmentation rate on this set demonstrates that the proposed segmentation scheme provides robust segmentation results without any manual tuning of parameters. The difference between the correct segmentation rate of set C compared to sets A and B is shown in the error analysis for set C. Almost a third of the errors were caused by wrong transcriptions of text lines. This due to the fact that no corrections of incorrect transcriptions have been made on set C.

Tab. 2 provides the error analysis for test set C. It shows the distribution of the detected segmentation errors over four types of errors. Component assignment errors usually result from small normalized connected components ending up at the "wrong" side of a word segmentation boundary. Fig. 8 gives an example of such an error. It has to be

Lines	Errors		Error category
59	122	(39%)	Component assignment
59	93	(30%)	Transcription
25	53	(17%)	Line extraction
31	42	(14%)	Filtering
162	310	(100%)	All

Table 2. Distribution of word segmentation errors on set C

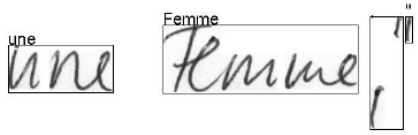

Figure 8. An example of a component assignment error

noted, that for this example two errors are counted since the bounding boxes for both the comma and the double quote were not correctly recognized.

Line extraction errors typically mix up some connected components or parts of connected components from either the line above or below. Those components are assigned to a word according to the word boundaries from the HMM based segmentation step. This frequently results in a bounding box which is larger than the minimal bounding box for the word under consideration. Most of the remaining errors are caused by either filtering out too many small components (periods or i-dots, for example) or not filtering some small components which are not part of any word and have been introduced by either the writer or the form scanning process.

Using the produced bounding box information for all correctly segmented text lines more than 25'000 isolated handwritten words could be automatically extracted from the 417 processed forms.

5 Conclusion and Future Work

We presented an automatic word segmentation scheme for off-line cursively handwritten text which achieved a correct word segmentation rate of 98% on a database for which ground truth on the level of complete lines exists. This segmentation scheme consists of two steps. In the first step the Viterby decoder of a Hidden Markov Model based recognition system is used in forced alignment mode. Using a normalized text line and its transcription, optimal word boundaries can be computed by the Viterby decoder. The second step uses these word boundaries to assign the connected components of the normalized line to the individual words and punctuation characters. The assigned connected components can then be used to compute the word bounding boxes for the original form image.

As a result of the presented work, we can now provide accurate bounding box information for over 25'000 handwritten words in the IAM database together with the corresponding word labels[5].

Using the provided segmentation ground truth the automatic extraction of isolated words becomes very easy. This helps to train and test recognizers for isolated handwritten words. Further applications of the segmentation ground truth lie in the automatic assessment of segmentation rates for the evaluation of different text segmentation algorithms or in the development of off-line recognition systems for unconstrained handwritten text.

For the future we plan to apply the presented automatic segmentation scheme on the more than 1000 remaining forms in the IAM database.

References

[1] D. Elliman and N. Sherkat. A truthing tool for generating a database of cursive words. In *6th Int. Conference on Document Analysis and Recognition, Seattle WA, USA*, pages 1255–1262, 2001.

[2] M. Garris. Design and collection of a handwritten sample image database. Technical report, NIST, 1992. Social Science Computer Review, volume 10, pages 196-214.

[3] V. Govindaraju, R. Srihari, and S. Srihari. Handwritten text recognition. In A. L. Spitz and A. Dengel, editors, *Document Analysis Systems*, pages 288–304. World Scientific, 1995.

[4] D. Guillevic and C. Suen. Cursive script recognition to the processing of bank cheques. In *Int. Conference on Document Analysis and Recognition 95, Montreal, Canada*, volume 1, pages 11–, 1995.

[5] J. Hull. A database for handwritten text recognition research. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(5):550–554, May 1994.

[6] S. Johansson, G. Leech, and H. Goodluck. *Manual of Information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital Computers.* Department of English, University of Oslo, Oslo, 1978.

[7] G. Kaufmann and H. Bunke. A system for the automated reading of check amounts - some key ideas. In *Proc. 3rd IAPR Workshop on Document Analysis Systems, Nagano, Japan*, pages 302 – 315, 1998.

[8] G. Kim, V. Govindaraju, and S. Srihari. An architecture for handwritten text recognition systems. *Int. Journal on Document Analysis and Recognition*, 2(1):37–44, July 1999.

[9] J. Mao, P. Sinha, and M. Mohiuddin. A system for cursive handwritten address recognition. In *14th Int. Conference on Pattern Recognition*, pages 1285–1287, Brisbane, Australia, 1998.

[10] U.-V. Marti and H. Bunke. A full English sentence database for off-line handwriting recognition. In *5th Int. Conference on Document Analysis and Recognition 99, Bangalore, India*, pages 705–708, 1999.

[11] U.-V. Marti and H. Bunke. Towards general cursive script recognition. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 203–212. World Scientific, 1999.

[5]The generated segmentation ground truth as well as the form images and line transcriptions are available from the authors upon request. Both the description of the segmentation ground truth as well as the line transcriptions are provided as ASCII files. The form images are currently stored in TIFF files.

[12] 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*, 15:65–90, 2001.

[13] U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for off-line handwriting recognition. *Int. Journal on Document Analysis and Recognition*, 5:39–46, 2002.

[14] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-9:pp. 62–66, 1979.

[15] A. Senior and A. Robinson. An off-line cursive handwriting recognition system. *IEEE Trans. on Patten Analysis and Machine Intelligence*, 20(3):309–321, Mar. 1998.