

# Handwriting Recognition of Whiteboard Notes

Marcus LIWICKI<sup>a</sup> and Horst BUNKE<sup>a</sup>

<sup>a</sup>*Department of Computer Science, University of Bern  
Neubrückestrasse 10  
CH-3012 Bern, Switzerland  
liwicki@iam.unibe.ch, bunke@iam.unibe.ch*

**Abstract.** This paper introduces a new system for processing on-line whiteboard notes. Notes written on a whiteboard is a new modality in handwriting recognition research that has received relatively little attention in the past. For the recognition we use an off-line HMM-recognizer, which has been developed in the context of our previous work. The recognizer is supplemented with methods for processing the on-line data and generating the images. The system consists of six main modules: on-line preprocessing, transformation to off-line data, off-line preprocessing, feature extraction, classification and post-processing. The recognition rate of the basic recognizer in a writer independent experiment is 59,5%. By applying state-of-the-art methods, such as optimizing the number of states and Gaussian components, and by including a language model we could achieve a statistically significant increase of the recognition rate to 64.3%.

## 1. Introduction

The domain of handwriting recognition has traditionally been divided into on-line and off-line recognition. In off-line recognition the text to be recognized is captured by a scanner and stored as an image, while in the on-line mode the handwriting is produced by means of an electronic pen or a mouse and acquired as a time-dependent signal. Good progress has been achieved in both off-line and on-line recognition (Plamondon and Srinhari, 2000). In this paper we consider a novel task, which is the recognition of text written on a whiteboard. A similar task has been considered in (Fink, Wienecke and Sagerer, 2001; Munich and Perona, 1996). However, while in (Fink, Wienecke and Sagerer, 2001; Munich and Perona, 1996) a video camera was used to capture the handwriting, we use the eBeam<sup>1</sup> interface which is based on infrared sensing. This system is easier to use than a video camera and it is less vulnerable to artifacts arising from poor lighting conditions, self-occlusion and low image resolution.

Our research is motivated by Smart Meeting Room applications (Moore, 2002; Reiter and Rigoll, 2004) where not only speech and video data of a meeting are recorded, but also notes written on a whiteboard are captured. In order to allow for indexing and browsing of the data collected during various meetings, the automatic recognition of whiteboard notes, i.e. their transcription into ASCII format, is needed.

Although the data output by the sensing device is in the on-line format, we use an HMM-based off-line recognizer in the work described in this paper. Our motivation is twofold. First, on-line data can be easily converted into the off-line format and secondly, we do have an off-line recognizer at our disposal that was developed in the context of our previous work (Marti and Bunke, 2001). Eventually we plan to combine the existing off-line recognizer with an on-line recognizer to be developed in the near future. From such a combination, an improved recognition performance can be expected (Velek, Jäger and Nakagawa, 2003; Vinciarelli and Perrone, 2003).

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed system. In Section 3 the main steps for preprocessing the data and the generation of the input for the basic recognizer are presented. Section 4 briefly describes the basic recognition system. The methods applied in the system to increase the recognition performance are described in Section 5. Experiments and results are presented in Section 6, and finally Section 7 draws some conclusions and gives an outlook for future work.

## 2. System Overview

The eBeam interface is used for recording the handwriting. It allows us to write on a whiteboard with a normal pen in a special casing, which sends infrared signals to a triangular receiver mounted in one of the corners of the whiteboard. The acquisition interface outputs a sequence of (x,y)-coordinates representing the location of the tip of the pen together with a time stamp for each location. The data is in xml-format and the frame rate of the recordings varies from 30 to 70 frames per second. An illustration is shown in Fig. 1.

The system described in this paper consists of six main modules (see Fig. 2): the on-line preprocessing, where noise in the raw data is reduced; the transformation, where the on-line data is transformed

---

<sup>1</sup> eBeam System by Luidia, Inc. - [www.e-Beam.com](http://www.e-Beam.com)



Figure 1. Illustration of the recording

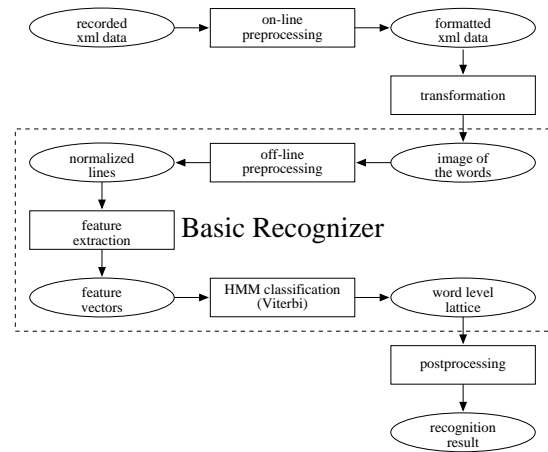


Figure 2. System overview

into off-line format; the off-line preprocessing, where various normalization steps take place; the feature extraction, where the normalized image is transformed into a sequence of feature vectors; the recognition, where the HMM-based classifier generates an n-best list of word sequences; and the post-processing, where a statistical language model is applied to improve the results generated by the HMM.

### 3. Preprocessing and On-line to Off-line Transformation

In writer-independent handwriting recognition, the preprocessing is a very important part because each writer has its individual writing style. The off-line preprocessing steps of (Marti and Bunke, 2001) are supplemented with additional on-line preprocessing operations to reduce the noise of the recording interface.

The recorded on-line data usually contain noisy points and gaps within strokes. In Fig. 3 examples of both cases are shown. In the word *await* a spurious point occurs that leads to the introduction of a large artifact, i.e. two long additional strokes. Furthermore, we observe in the first line that there are many gaps within the word, which are caused by loss of data.

Thus two on-line preprocessing steps are applied to the data, to recover from artifacts of this kind. Let  $p_1, \dots, p_n$  be the points of a given stroke and  $q_1$  be the first point of the succeeding stroke, if any. First, if the distance between two consecutive points  $p_i, p_{i+1}$ , is larger than a fixed threshold, one of the points is deleted. To decide which point has to be deleted, the number of points within a small neighborhood of  $p_i$  and  $p_{i+1}$  are determined, and the point with a smaller number of neighbors is deleted. To recover from artifacts of the second type, we check if the distance between the timestamps of  $p_n$  and  $q_1$  is under a fixed threshold. If the condition is true the strokes are merged into one stroke. Note that the thresholds used for preprocessing depend on the recording environment. They can be derived from the frame-rate and the width of the pen strokes, or by scaling the average distance between succeeding timestamps in the recorded data. An example of the results of these preprocessing steps is shown in Fig. 4. Obviously, now the handwriting is of much better quality.

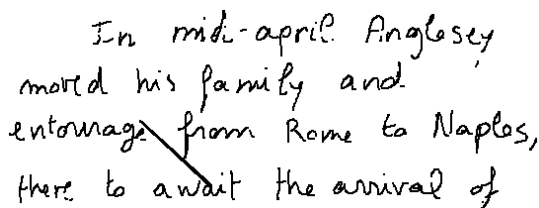


Figure 3. Recorded text

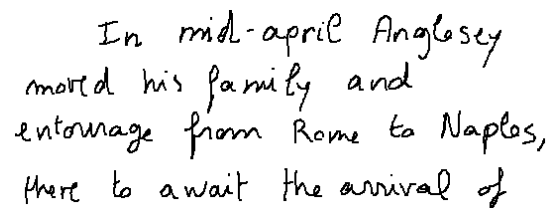


Figure 4. Text after removing noise

Since the preprocessed data is still in the on-line format, it has to be transformed into an off-line image, so that it can be used as input for the off-line recognizer. The recognizer was originally designed for the off-line IAM-Database (Marti and Bunke, 2002) and optimized on gray-scale images scanned with a resolution of 300 dpi. To get good recognition results in the considered application, the produced images should be similar to these off-line images. Consequently the following steps are applied to generate the images. First, all consecutive points within the same stroke are connected. This results in one line segment per stroke. Then the lines are dilated to a width of eight pixels. The center of each line is colored black and the pixels are getting lighter towards the periphery. Fig. 5 shows an example of a generated image.

Compared to Figs. 3 and 4 the handwriting looks more similar to the IAM-Database (see Fig. 6). In general, the realistic generation of off-line data is a quite complex problem. (Velek et al., 2002) proposes methods to create images that look even more similar to scanned images. In the experiments reported in (Velek et al., 2002) the recognizer was trained and tested on generated images, and the best performance has been achieved using a constant thickness. During our experiments the recognition rate increased when we supplemented this simple approach with the generation of different gray values.



**Figure 5.** Generated gray-scale image



**Figure 6.** Image of the IAM-Database (produced by a writer different from Fig 5)

#### 4. The Basic Recognizer

The basic recognizer is a Hidden Markov Model (HMM) based cursive handwriting recognizer similar to the one described in (Marti and Bunke, 2001). For the purpose of completeness we include a brief description here. For any further details the reader is referred to (Marti and Bunke, 2001).

The basic recognizer takes, as an input unit, a complete text line, which is first normalized with respect to skew, slant, writing width and baseline location. Normalization of the baseline location means that the body of the text line (the part which is located between the upper and lower baselines), the ascender part (located above the upper baseline), and the descender part (below the lower baseline) will be vertically scaled to a predefined size each. Writing width normalization is performed by a horizontal scaling operation, and its purpose is to scale the characters so that they have a predefined average width.

To extract the feature vectors from the normalized images, a sliding window approach is used. The width of the window is one pixel and nine geometrical features are computed at each window position. Thus an input text line is converted into a sequence of feature vectors in a 9-dimensional feature space. The nine features correspond to the following geometric quantities. The first features represent the average gray value of the pixels in the window and their center of gravity. Furthermore, the second order moment in vertical direction is taken. In addition to these global features, the location of the uppermost and lowermost black pixel are used. Their positions and their gradients, determined by using the neighboring windows, are taken. Feature number eight is the number of black-white transitions between the uppermost and lowermost pixel in an image column. Finally, the proportion of black pixels to the number of pixels between these two points is used. For a detailed description of the features see (Marti and Bunke, 2001).

An HMM is built for each of the 58 characters in the character set, which includes all small and capital letters and some other special characters, e.g. punctuation marks. In all HMMs the linear topology is used, i.e. there are only two transitions per state, one to itself and one to the next state. In the emitting states, the observation probability distributions are estimated by mixtures of Gaussian components. In other words, continuous HMMs are used. The character models are concatenated to represent words and sequences of words. For training, the Baum-Welch algorithm (Dempster, Laird and Rubin, 1977) is applied. In the recognition phase, the Viterbi algorithm (Forney, 1973) is used to find the most probable word sequence. As a consequence, the difficult task of explicitly segmenting a line of text into isolated words is avoided, and the segmentation is obtained as a byproduct of the Viterbi decoding applied in the recognition phase. The output of the recognizer is a sequence of words. In the experiments described in Section 6, the recognition rate will always be measured on the word level.

The inclusion of a language model, which is the postprocessing step illustrated in Fig. 2, will be described in the next section.

#### 5. Optimizing the Basic Recognizer

Recently, some methods for increasing the performance of an HMM-based recognizer have been proposed (Zimmermann and Bunke, 2004; Günter and Bunke, 2004; Zimmermann and Bunke, 2002). In the system presented in this paper, these methods are adopted. We will describe the main ideas of the optimization strategies in this section.

As already mentioned in Section 4., one HMM is trained for each character. In the original system described in (Marti and Bunke, 2001) a constant number of states was defined for all characters. However, in (Zimmermann and Bunke, 2002) it was shown that the recognition rate can be improved by letting each character having an individual number of states. Optimizing the number of states of each individual character was accomplished by the Bakis method (Bakis, 1976), which sets the length of each character-HMM to a fraction of the average character width found in the training set. Since the system described in this paper creates input images that are similar to the input images used in (Zimmermann and Bunke, 2002) (see Figs. 5 and 6), we have adopted this approach here.

**Table 1.** Performance of basic and optimized systems for each combination of the cross-validation on the validation sets

combination	$c0$	$c1$	$c2$	$c3$	$c4$	<i>avg.</i>
validation set	$s1$	$s2$	$s3$	$s4$	$s0$	
basic system	61.2	61.4	66.5	52.2	56.4	59.5
1st opt. sys.	64.1	62.3	66.9	52.1	59.0	60.9
2nd opt. sys.	66.8	66.9	71.2	57.1	65.8	65.6

**Table 2.** Performance of basic and optimized systems for each combination of the cross-validation on the test sets

combination	$c0$	$c1$	$c2$	$c3$	$c4$	<i>avg.</i>
test set	$s0$	$s1$	$s2$	$s3$	$s4$	
basic system	60.0	62.7	59.8	65.1	50.3	59.6
1st opt. sys.	60.4	61.6	61.2	63.9	52.1	59.8
2nd opt. sys.	66.5	64.1	64.7	70.5	55.6	64.3

In (Günter and Bunke, 2004) it is pointed out that the number of Gaussians and training iterations have an effect on the recognition results. There, three strategies for jointly optimizing the number of Gaussians and training iterations are proposed. The strategy that obtained the best performance in (Günter and Bunke, 2004) has been adopted for the system described in this paper. Under this strategy, the number of Gaussian components is incremented stepwise by one. In each step, the optimal number of training iterations is found by testing the performance on a separate validation set. For the next step the best performing system is chosen. It has been observed that the optimal number of Gaussian components increases with the amount of data since more variations are encountered (Günter and Bunke, 2004). Thus the system proposed in this paper has been trained with up to 20 Gaussian components and the classifier that performed best on the validation set has been taken as the final one for the respective experiment described in Section 6.

Since the system proposed in this paper is performing handwritten text recognition on text lines and not only on single words, it is reasonable to integrate a statistical language model. In (Zimmermann and Bunke, 2004) it was shown that by means of such an integration the performance of the recognizer can be significantly improved. In the method described in (Zimmermann and Bunke, 2004), two parameters have been used, the Grammar Scale Factor (GSF) and the Word Insertion Penalty (WIP). The first parameter weights the influence of the language model against the optical model, while the latter parameter can prevent the system from oversegmentation resp. undersegmentation. For more details on the effects of these parameters see (Zimmermann and Bunke, 2004). In an attempt to optimize the system described in this paper we have also included a statistical language model with these two parameters.

## 6. Experiments and Results

The overall amount of the recorded data is 6,204 words in 1,258 lines from 20 different writers. Each writer wrote approximately the same number of words. The data set was randomly divided into five disjoint sets of approximately equal size (sets  $s_0, \dots, s_4$ ). On these sets, 5-fold cross validation was performed in the following way (combinations  $c_0, \dots, c_4$ ). For  $i = 0, \dots, 4$ , sets  $s_{i \oplus 2}, s_{i \oplus 3}$  and  $s_{i \oplus 4}$  were taken for training the recognizer, set  $s_{i \oplus 1}$  was used as a validation set, i.e. for optimizing the parameters in the optimization steps, and set  $s_i$  was used as a test set for measuring the system performance. Each of the sets consists of the data of four writers and no writer appears in more than one set. Consequently, writer-independent recognition experiments were conducted. The word-dictionary includes exactly those 2,337 words that occur in the union of all of the five sets. The language model was generated from the LOB-Corpus (Johansson, 1986), which contains 500 printed texts of about 2,000 words each.

The basic recognizer has been trained with 32 iterations. After every 4th iteration, a splitting operation had been applied, to increase the number of Gaussians by one. The average recognition rate of this recognizer is 59,54% on the five validation sets and 59.59% on the five test sets.

An enhancement of the training procedure used for the basic recognizer is obtained if the number of Gaussians and training iterations are simultaneously optimized, as described in Section 5. For each number of Gaussians, the optimal number of training iterations is determined on the validation set. By means of this procedure, the average recognition rate on the validation set could be increased up to 60.88%. On the test set only a slightly improvement from 59.59% to 59.83% was achieved. This indicates that the number of eight Gaussian components for the basic recognizer is already a good choice.

In Section 5 the inclusion of a language model has been described. To find the best values for the parameters GSF and the WIP, we calculated the performance on the validation set for different combinations, i.e. GSF varied from 0 to 90 and WIP from -150 to 150. By including the resulting bigram language model, the average recognition rate could be further increased up to 65.56% on the validation set and 64.27% on the test set. This improvement is statistically significant.

A summary of all experimental results is provided in Tab. 1 (validation set) and Tab. 2 (test set). The performance of the optimized system on the test sets varies from 55.6% on set  $s_4$  to 70.5% on set  $s_3$ . This large variation is due the fact that each set includes data from only four writers and the writing style of some writers is very different from all other styles. We expect to overcome problems of this kind by using larger data sets.

## 7. Conclusions and Future Work

In this paper we presented a handwriting recognition system for whiteboard notes. Notes written on a whiteboard is a new modality in handwriting recognition research that has received relatively little attention in the past. The proposed system is based on an off-line handwriting recognition system, which has been developed in our previous work. We added some on-line preprocessing methods to recover from artifacts produced by the recording system. To achieve a better recognition performance, we applied some state-of-the-art optimization methods.

Obviously the present recognition rate is still too low to be practically useful. However, there is significant potential for further improvement, for example, by enlarging the training set. We also plan to develop an on-line recognizer and combine it with the off-line recognition system described in this paper. To further enhance the practical usefulness of the system in a smart meeting room scenario, recognition of mathematical formulas (Tapia and Rojas, 2003) and tables (Zanibbi, Blostein and Cordy, 2004) will be considered.

## Acknowledgements

This work was supported by the Swiss National Science Foundation program “Interactive Multimodal Information Management (IM)<sup>2</sup>” in the Individual Project “Scene Analysis”, as part of NCCR. The authors thank Dr. Urs-Viktor Marti and Dr. Matthias Zimmermann for providing the recognition system and the IAM-Database, respectively and Dr. Simon Günter for his valuable comments and advice. We also thank Dr. Samy Bengio, Dr. Darren Moore and Joanne Moore of IDIAP for valuable assistance in the collection of the database used in the experiments described in this paper.

## References

- Bakis, R. 1976. Continuous speech recognition via centisecond acoustic states. In *Proc. 91st Meeting of the Acoustic Society in America*.
- Dempster, A.P., N.M. Laird and D.B. Rubin. 1977. “Maximum Likelihood from Incomplete Data via the EM Algorithm.” *Journal of Royal Statistical Society B* 39(1):1–38.
- Fink, G. A., M. Wienecke and G. Sagerer. 2001. Video-based on-line handwriting recognition. In *Proc. 6th ICDAR*. pp. 226–230.
- Forney, G. D. 1973. The Viterbi algorithm. In *Proc. IEEE*. Vol. 61 pp. 268–278.
- Günter, S. and H. Bunke. 2004. “HMM-based handwritten word recognition: on the optimization of the number of states, training Iterations and Gaussian components.” *Pattern Recognition* 37:2069–2079.
- Johansson, S. 1986. *The tagged LOB Corpus: User’s Manual*. Norwegian Computing Centre for the Humanities, Norway.
- Marti, U.-V. and H. Bunke. 2001. “Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system.” *IJPRAI* 15:65 – 90.
- Marti, U.-V. and H. Bunke. 2002. “The IAM-database: an English sentence database for offline handwriting recognition.” *IJDAR* 5:39 – 46.
- Moore, D. 2002. The IDIAP Smart Meeting Room. Technical report IDIAP-Com.
- Munich, M.E. and P. Perona. 1996. Visual input for pen-based computers. In *Proc. 3rd ICPR*. pp. 33–37.
- Plamondon, Rjean and Sargur N. Srinhari. 2000. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. In *IEEE TPAMI*. Vol. 22 pp. 63–84.
- Reiter, S and G Rigoll. 2004. Segmentation and Classification of Meeting Events using Multiple Classifier Fusion and Dynamic Programming. In *Proc. 17th ICPR*. pp. 434–437.
- Tapia, E. and R. Rojas. 2003. Recognition of On-line Handwritten Mathematical Formulas in th E-Chalk System. In *Proc. 7th ICDAR*. pp. 980– 984.
- Velek, O., C.-L. Liu, S. Jaeger and M. Nakagawa. 2002. An Improved Approach to Generating Realistic Kanji Character Images from On-Line Characters and its Benefit to Off-line Recognition Performance. In *Proc. 16th ICPR*. pp. 588– 591.
- Velek, O., S. Jäger and M. Nakagawa. 2003. Accumulated-Recognition-Rate Normalization for Combining Multiple On/Off-Line Japanese Character Classifiers Tested on a Large Database. In *Proc. 4th Multiple Classifier Systems*. pp. 196–205.
- Vinciarelli, A. and M. Perrone. 2003. Combining Online and Offline Handwriting Recognition. In *Proc. 7th ICDAR*. pp. 844– 848.
- Zanibbi, R., D. Blostein and J.R. Cordy. 2004. “A Survey of Table Recognition: Models, Observations, Transformations and Inferences.” *IJDAR* 7(1):1–16.
- Zimmermann, M. and H. Bunke. 2002. Hidden Markov Model Length Optimization for Handwriting Recognition Systems. In *Proc. 8th IWFHR*. pp. 369 – 374.
- Zimmermann, M. and H. Bunke. 2004. Optimizing the Integration of a Statistical Language Model in HMM-based Offline Handwritten Text Recognition. In *Proc. 17th ICPR*. pp. 541 – 544.