

International Journal of Pattern Recognition and Artificial Intelligence
© World Scientific Publishing Company

HANDWRITING RECOGNITION OF WHITEBOARD NOTES - STUDYING THE INFLUENCE OF TRAINING SET SIZE AND TYPE

MARCUS LIWICKI and HORST BUNKE

*Institut für Informatik und angewandte Mathematik
Universität Bern, Neubrückestrasse 10, CH-3012 Bern, Switzerland
{liwicki,bunke}@iam.unibe.ch
<http://www.iam.unibe.ch/~liwicki>*

This paper presents a system for the recognition of 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 is supplemented with methods for processing the on-line data and generating off-line images. The system consists of six main modules: on-line preprocessing, transformation of on-line to off-line data, off-line preprocessing, feature extraction, classification and post-processing. The recognition rate of our 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%. To further improve the system performance we increased the size of the training set. For that we investigated two different strategies. First we used another existing database of off-line handwritten text. Second we used a recently collected whiteboard database, called the IAM-OnDB. By means of these strategies the recognition rate could be further increased up to 68.5%.

Keywords: Cursive Handwritten Text Recognition, eBeam Whiteboard Interface, Statistical Language Model, Hidden Markov Model (HMM), MAP-Adaptation

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 recognition¹⁷. 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 refs. 4, 16. However, while in refs. 4, 16 a video camera was employed to capture the handwriting, we use the eBeam^a interface which is based on infrared sensing. This system is easier

^aeBeam System by Luidia, Inc. - www.e-Beam.com

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^{15,18,25} where not only speech and video data of a meeting are recorded, but also notes written on a whiteboard are captured. In a smart meeting room we typically find multiple acquisition devices that are used to record a meeting. In order to allow for retrieval of the meeting data by means of a browser, semantic information needs to be extracted from the raw sensory data, such as transcription of speech and recognition of persons in video images. Whiteboards are commonly used in meeting rooms. Hence capture and automatic transcription of handwritten notes on a whiteboard are essential tasks in a smart meeting room application.

Although the data output by the whiteboard 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. Firstly 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 work¹³. 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^{21,24}.

A preliminary version of this article appeared in Ref. 11. In the present paper we describe a significantly enhanced version of the system. Novel recognizers were developed by enlarging the training set in two ways. First we used another existing database of handwritten text, which includes handwriting samples different from whiteboard data and did an adaptation of the recognizer. Second we used a recently collected large database of handwritten whiteboard data, called IAM-OnDB¹². This database is publicly available on the World Wide Web^b. The work described in this paper is loosely related to work reported in Ref. 10. While the current paper takes the expansion of the training set size into account, similarly to Ref. 10, it additionally investigates the type of the supplementing training data. It further combines all expansion experiments to study the effect of using one large training set consisting of all available training data.

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed system and introduces the main steps for preprocessing the data, generating the input images for the recognizer, and recognizing the handwritten text. In Section 3 the main ideas for enhancing the training data are presented. Experiments and results are described in Section 4, and finally Section 5 draws some conclusions and gives an outlook for future work.

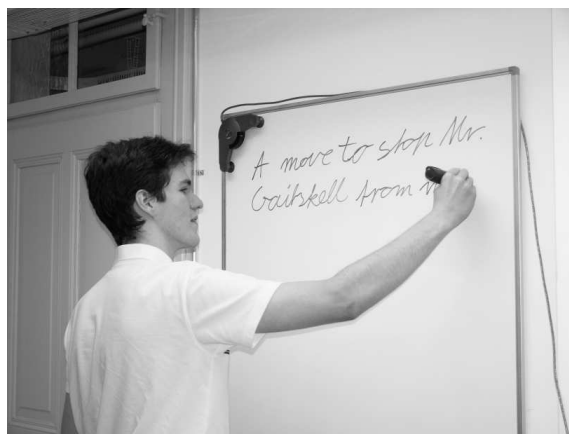


Fig. 1. Illustration of the recording

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 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 an 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.

2.1. Preprocessing

In writer-independent handwriting recognition, the preprocessing is a very important part because each writer has his or her individual writing style. The off-line preprocessing steps of Ref. 13 are supplemented with additional on-line preprocessing operations to reduce the noise introduced by the recording interface. The recorded on-line data usually contain noisy points and gaps within strokes. In Fig. 3

^b<http://www.iam.unibe.ch/~fki/iamondb/>

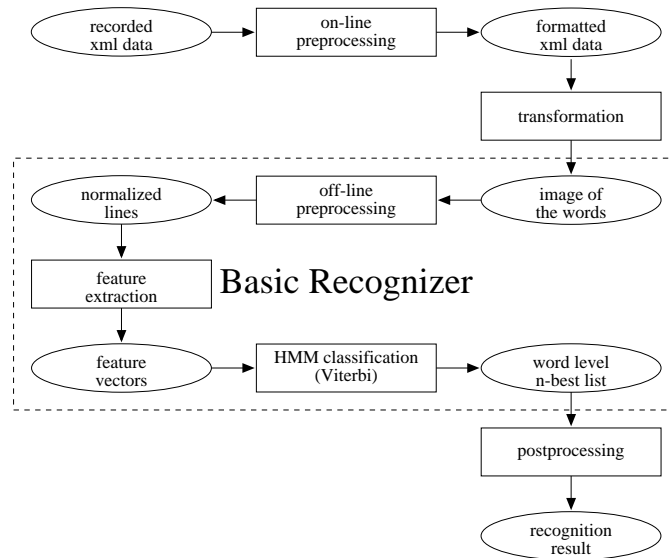
4 *M. Liwicki and H. Bunke*

Fig. 2. System overview

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 the 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^c. An example of the results of these preprocessing steps is shown in Fig. 4. Obviously, now the handwriting is of much better quality.

^cIn our experiments we used $2 * m$, where m is the average value of the time intervals, or distances, respectively. This is reasonable, because at $2 * m$, there are distinct gaps in the histogram of all values.

In mid-april Anglesey
moved his family and
entourage from Rome to Naples,
there to await the arrival of

Fig. 3. Recorded text

In mid-april Anglesey
moved his family and
entourage from Rome to Naples,
there to await the arrival of

Fig. 4. Text after removing noise

2.2. On-line to off-line transformation

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-Database¹⁴ 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. Ref. 22 proposes methods to create images that look even more similar to scanned images. In the experiments reported in Ref. 22 the recognizer was trained and tested on computer 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.

In mid-april Anglesey

Fig. 5. Generated gray-scale image

In mid-april Anglesey

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

2.3. Basic recognizer

The basic recognizer is a Hidden Markov Model (HMM) based cursive handwriting recognizer similar to the one described in Ref. 13. For the purpose of completeness we include a brief description here. For any further details the reader is referred to Ref. 13.

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 Ref. 13.

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 algorithm³ is applied. In the recognition phase, the Viterbi algorithm⁵ 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 4, 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 Section 2.5.

2.4. *Optimizing the basic recognizer*

Recently, some methods for increasing the performance of HMM-based recognizers have been proposed^{7,28,29}. 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 2.3, one HMM is trained for each character. In

the original system described in Ref. 13 a constant number of states was defined for all characters. However, in Ref. 28 it was shown that the recognition rate can be improved by letting each character have an individual number of states. Optimizing the number of states of each individual character was accomplished by the Bakis method¹, 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 Ref. 28 (see Figs. 5 and 6), we have adopted this approach here.

In Ref. 7 it has been 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 Ref. 7 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 encountered⁷. 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 4.

2.5. Including a language model

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 this paper we consider bigram language models. Bigram language models represent a special case of the more general statistical n -gram language models¹⁹.

N -gram language models are based on the observation that we are often able to guess the next word when we are reading a given text. In other words, the probability of a word is highly depending on the previous text. In the case of bigram language models the previous text is approximated by the last word and the dependency is modeled by the probability $p(w_i|w_{i-1})$, where w_i represents the considered word and w_{i-1} stands for the previous word. The probability $p(W)$ of a text line $W = (w_1, \dots, w_n)$ can then be computed as follows:

$$p(W) = p(w_1) \left(\prod_{i=2}^n p(w_i|w_{i-1}) \right) \quad (1)$$

The most likely word sequence $\hat{W} = (w_1, \dots, w_m)$ for a given observation sequence X is computed in the following way²⁹:

$$\hat{W} = \underset{W}{\operatorname{argmax}} \log p(X|W) + \alpha \log p(W) + m \beta \quad (2)$$

their sale being by the

GSF	WIP	Recognition result
0	-40	their sale being by by
0	0	their sale being by in E
15	0	their sale being long
15	20	their sale being by the
90	10	their sale being long
90	130	their sale be in a by this ,

Fig. 7. Influence of the parameters GSF and WIP

According to Eq. 2 the optical model $p(X|W)$, which is the result of the HMM decoding, is combined with the likelihood $p(W)$ obtained from the language model. Because the HMM system and the language model produce only approximations of probabilities, two additional parameters α and β are necessary to compensate the deficiencies and to control the integration of the language model. The parameter α is called Grammar Scale Factor (GSF) and weights the influence of the language model against the optical model. The term Word Insertion Penalty (WIP) is used for the parameter β which prevents the system from oversegmentation resp. undersegmentation. That is, the larger the value of β is, the higher is the system's tendency to split a text into many short words. Figure 7 illustrates the influence of these two parameters on an example text line. In an attempt to optimize the system described in this paper we have included a statistical language model with these two parameters.

3. Enhancing the Training Data

Our first experiments with the system described in Section 2 are based on a rather small database. In fact, the recognition rate of these experiments, described in Section 4.1, is quite low. But there is significant potential for further improvement. It is a well-known fact that enlarging the size of the training set has an influence on the recognition performance. However, collecting more data is very time consuming, because the whiteboard is not portable and can be used by only a single writer at a time. For this reason we also investigate another approach in this article, where we use data from a large existing database of off-line handwritten sentences¹⁴ to augment the training set. Furthermore the use of the large new on-line handwrit-

ing database IAM-OnDB¹² is described. We will compare these two approaches in Section 4 with each other.

3.1. Enhancing with a different database

Since the off-line images generated from the whiteboard data are similar to the images of the IAM-Database (see Section 2.2), the IAM-Database¹⁴ can be used to enhance the small dataset of whiteboard recordings. The IAM-Database includes over 1,500 scanned forms of texts written on normal paper by more than 650 writers. For our experiments a subset containing about 18,000 words in 1,993 text lines produced from 400 different writers was taken.

We have investigated two different approaches to combining the training set originally used in Ref. 11 with a subset of the IAM-Database. First an HMM-recognizer trained on the IAM-Database is taken and adapted with the Maximum A Posteriori (MAP) estimation method⁶. Adaptation methods are usually used for writer adaptation in writer dependent systems²³, but have been successfully applied in writer independent tasks as well². MAP estimation has been chosen for adaptation because it produced the best recognition results in Ref. 23 on adaptation sets of larger size. In the second approach the HMM-recognizer is trained on the union of the selected IAM-Database subset and the whiteboard data.

HMM adaptation is a method to adjust the model parameters θ of a given background model (the HMMs trained on the IAM-Database in our case) to the parameters θ_{ad} of the adaptation set of observations O (the training set of the whiteboard data). The aim is to find the vector θ_{ad} which maximizes the *posterior* distribution $p(\theta_{ad}|O)$:

$$\theta_{ad} = \operatorname{argmax}_{\theta} (p(\theta|O)) \quad (3)$$

Using Bayes theorem $p(\theta|O)$ can be written as follows:

$$p(\theta|O) = \frac{p(O|\theta)p(\theta)}{p(O)} \quad (4)$$

where $p(O|\theta)$ is the likelihood of the HMM with parameter set θ and $p(\theta)$ is the *prior* distribution of the parameters. When $p(\theta) = c$, i.e. when the *prior* distribution does not give any information about how θ is likely to be, Maximum Likelihood Linear Regression (MLLR⁹) can be performed. If the prior distribution is informative, i.e. $p(\theta)$ is not a constant, the adapted parameters can be found by solving the equation

$$\frac{\partial}{\partial \theta} (p(O|\theta)p(\theta)) = 0 \quad (5)$$

This minimizes the Bayes risk over the adaptation set and can be done with Maximum A Posteriori (MAP) estimation, which is also called Bayesian Adaptation. As described in Ref. 23, it is feasible to adopt only the Gaussian means μ_{jm} (where m refers to the actual state and j is the index of the considered mixture in state

10 *M. Liwicki and H. Bunke*

m) of the parameters θ of each HMM. The use of conjugate priors then results in a simple adaptation formula:

$$\hat{\mu}_{jm} = \frac{N_{jm}}{N_{jm} + \tau} \bar{\mu}_{jm} + \frac{\tau}{N_{jm} + \tau} \mu_{jm} \quad (6)$$

where $\hat{\mu}_{jm}$ is the new and $\bar{\mu}_{jm}$ the old mean of the adaptation data, μ_{jm} is the mean of the background model, and N_{jm} is the sum of the probabilities of each observation in the adaptation set being emitted by the corresponding Gaussian. After each iteration the values of $\hat{\mu}_{jm}$ are used in the Gaussians, which leads to new values of $\bar{\mu}_{jm}$ and N_{jm} in Eq. (6). This procedure is repeated until the change in the parameters falls below a predefined threshold. The parameter τ weights the influence of the background model on the adaptation data. Whereas it has been set empirically in Ref. 23, it is optimized on a validation set in this paper. MAP estimation has been chosen for adaptation since it produced the best recognition results in Ref. 23 on adaptation sets of larger size. This is due to the fact that it adapts each Gaussian separately.

Our second approach to training set enhancement is much simpler. It just uses data from both the IAM-Database and the whiteboard data collection, i.e. the data of the IAM-Database and the training set of the whiteboard data are combined into one large training set. This set is then used for training the HMM recognition system. The validation set for optimizing the parameters is taken only from the whiteboard data. Thus, in contrast to the adaptation, the training of the recognizer is optimized on the whiteboard data. This strategy consumes more time than the adaptation method because the training is performed on a larger dataset.

3.2. *The new on-line database*

Recently the IAM-OnDB, a new large on-line handwriting database consisting of samples acquired on a whiteboard, has been released¹². This database contains 86,272 word instances from a 11,050 word dictionary written down in 13,040 text lines by 221 writers. Analogously to mixed training on the small whiteboard data set and the IAM-Database, we conducted recognition experiments using these samples for supplementing the training set.

4. Experiments and Results

In this section a large set of experiments is presented to investigate the influence of the training set size as well as the type of the training data on the recognizer's behavior. At first we will describe the baseline experiments and the effect of the optimization steps discussed in Sections 2.4 and 2.5. Then the training set enhancement strategies are evaluated. At the end a summary of all the experiments is provided.

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

combination	$c0$	$c1$	$c2$	$c3$	$c4$	avg.
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

4.1. Small whiteboard data set

The amount of the recorded data in our first experiments (small set) 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 data from 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-Corpus⁸. 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 59.5% on the five validation sets and 59.6% 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 2.4. 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 to 60.9%. On the test set only a slightly improvement from 59.6% to 59.8% was achieved. This indicates that eight Gaussian components for the basic recognizer is already a good choice.

In Section 2.5 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 64.3% on the test set. This improvement is statistically significant. A summary of the first experimental results on the test set is provided in Tab. 1. Note that this is the baseline experiment, where only the small whiteboard dataset is used for training. For the readers' convenience

Table 2. Optimized values for the constants for the combination $c\theta$

parameter	value
Number of Gaussians	16
GSF for Language Model	25
WIP	10
τ for MAP estimation	0.1

the optimal values of the parameters in our experiments are listed in Tab. 2 for the combination $c\theta$. However, these parameters depend on the amount of training data and the quality of the language model. Hence they were all optimized on validation sets before applying the resulting recognition system to the test set.

4.2. *Enhancing with data from the IAM-Database*

For enhancing the training data a subset of the IAM-Database¹⁴ containing about 18,000 words in 1,993 text lines produced from 400 different writers was taken. The background model for the adaptation was trained on the data of 320 writers. For the resulting recognizer the number of Gaussians were optimized on the validation set consisting of the remaining 80 writers. The performance of this recognizer is 54.9% on the whiteboard data (without using a language model). As described in Section 3.1 this recognizer was adapted with the training sets from the recorded whiteboard data. To find the best value of parameter τ for the MAP estimation we calculated the performance on the validation set of the whiteboard data for different values of τ . This optimization has been done for each combination of the cross validation separately. The average recognition rate on the test sets is 64.6% without a language model. By including a bigram language model, the average performance could be increased to 67.5%. Table 2 also shows the optimal value of τ for the example of combination $c\theta$. The influence of the background model is small because of the large amount of adaptation data.

By training on the IAM-Database and the small whiteboard data set simultaneously, as described in the end of Section 3.1, the performance is even higher. The recognition rate on the test sets is 65.3% without a language model. It increases up to 68.5% when the language model is included. We note that for both the non-optimized and the optimized system a substantial improvement of the recognition rate is achieved by the proposed methods.

4.3. *Experiments on the IAM-OnDB*

To investigate the effect of a growing amount of whiteboard training data, we increased the training set size in two steps. First we used a larger set produced by 50 writers and tested the recognizer (medium set). Second we used the full IAM-

Table 3. Overview of all experiments (LM - language model)

combination	<i>without LM</i>	<i>including a LM</i>
basic system	59.8	64.3
system adapted from IAM-DB	64.6	67.5
training on mixed data with IAM-DB	65.3	68.5
training on medium set of IAM-OnDB	60.4	64.8
training on large set of IAM-OnDB	61.0	66.4
training on IAM-DB and IAM-OnDB	65.5	68.6

OnDB for the experiments. The experimental conditions (5-fold cross validation using a training, a validation, and a test set) were always the same as for the small data set.

In the 50-writer experiments there were texts of 30 writers in addition to the small set. These texts were added to each training set s_i . We validated the optimization parameters on the same validation sets and tested the performance on the same test sets as in the experiments with the small set. The average recognition rate on the test sets is 60.4% without a language model. It increases to 64.8% by including a language model.

In the large experiments all data from the writers that do not appear in the test sets was used for training. There the average recognition rate is 61.0% on the test sets. By integrating a language model the performance could be increased to 66.4% on the test sets.

Having both IAM-DB and IAM-OnDB available led to the idea of combining both databases into one very large training set. The resulting recognition rate is 65.5% without and 68.6% with a language model. Obviously, this is the best recognition rate described in the experiments reported in this paper. Table 3 gives a summary of all our experiments.

In our experiments we observed that the enhancement with the off-line IAM-DB has been slightly better than using the IAM-OnDB. This may be surprising at first glance. However, a possible reason is that in the experiments with the IAM-DB more than 300 writers have been used, while it have been less than 200 in the experiments including the IAM-OnDB. The good performance of the system trained on data from the off-line IAM-DB confirms the statement that the generated images of the whiteboard data are very similar to the images of the IAM-DB.

5. 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 re-

search 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 state-of-the-art methods, such as optimization of the number of states and Gaussian mixture components, and by inclusion of a statistical language model.

To further improve the performance, we enhanced the training data in several ways. First we used additional data from the IAM-Database. The recognition rate could be significantly increased by 4.2% to 68.5% by training on mixed training data. Second we used a larger database of handwritten notes acquired on a whiteboard, namely the IAM-OnDB. With the second database, the recognition rate could also be increased, by 2.1% to 66.4%. This increase is statistically significant as well. Finally we combined both database into one large training set. A recognition rate of 68.6% was achieved, which is the best one obtained in all experiments reported in this paper.

In future research we plan to develop an on-line recognizer and combine it with the off-line recognition system described in this paper. We also plan to improve the normalization methods, because whiteboard data differs from usual document data in that the geometrical distortion of the handwriting is larger²⁶. To further enhance the practical usefulness of the system in a smart meeting room scenario, recognition of mathematical formulas²⁰ and tables²⁷ need to be considered.

Acknowledgments

This work was supported by the Swiss National Science Foundation program “Interactive Multimodal Information Management (IM)²” 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. The authors thank all volunteers who took the time for participating in the recording sessions. We also thank Dr. Darren Moore for providing us basic driver software for the eBeam system.

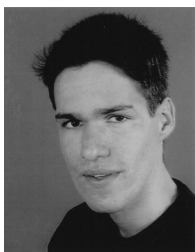
References

1. R. Bakis. Continuous speech recognition via centisecond acoustic states. In *Proc. 91st Meeting of the Acoustic Society in America*, 1976.
2. A. Brakensiek and G. Rigoll. Handwritten address recognition using hidden markov models. In A. Dengel et al., editor, *Reading and Learning*, volume 2956 of *LNCIS*, pages 103–122. Springer, 2004.
3. A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, 39(1):1–38, 1977.
4. G. A. Fink, M. Wienecke, and G. Sagerer. Video-based on-line handwriting recogni-

- tion. In *Proc. 6th Int. Conf. on Document Analysis and Recognition*, pages 226–230, 2001.
5. G. D. Forney. The Viterbi algorithm. In *Proc. IEEE*, volume 61, pages 268–278, 1973.
 6. J.-L. Gauvain and C.-H. Lee. Map estimation of continuous density HMM: Theory and applications. In *Proc. of DARPA Speech and Natural Language Workshop*, pages 272–277, 1992.
 7. S. Günter and H. Bunke. HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. *Pattern Recognition*, 37:2069–2079, 2004.
 8. S. Johansson. *The tagged LOB Corpus: User's Manual*. Norwegian Computing Centre for the Humanities, Norway, 1986.
 9. C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech and Language*, 9:171–185, 1995.
 10. M. Liwicki and H. Bunke. Enhancing training data for handwriting recognition of whiteboard notes with samples from a different database. In *Proc. 8th Int. Conf. on Document Analysis and Recognition*, volume 2, pages 550–554, 2005.
 11. M. Liwicki and H. Bunke. Handwriting recognition of whiteboard notes. In *Proc. 12th Conf. of the International Graphonomics Society*, pages 118–122, 2005.
 12. 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.
 13. 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.
 14. U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *Int. Journal on Document Analysis and Recognition*, 5:39–46, 2002.
 15. D. Moore. The IDIAP smart meeting room. Technical report, IDIAP-Com, 2002.
 16. M.E. Munich and P. Perona. Visual input for pen-based computers. In *Proc. 3rd Int. Conf. on Pattern Recognition*, pages 33–37, 1996.
 17. Rjean Plamondon and Sargur N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):63–84, 2000.
 18. S Reiter and G Rigoll. Segmentation and classification of meeting events using multiple classifier fusion and dynamic programming. In *Proc. 17th Int. Conf. on Pattern Recognition*, pages 434–437, 2004.
 19. Ronald Rosenfeld. Two decades of statistical language modeling: Where do we go from here? In *Proc. IEEE 88*, volume 88, pages 1270–1278, 2000.
 20. E. Tapia and R. Rojas. Recognition of on-line handwritten mathematical formulas in the e-chalk system. In *Proc. 7th Int. Conf. on Document Analysis and Recognition*, pages 980–984, 2003.
 21. O. Velek, S. Jäger, and M. Nakagawa. Accumulated-recognition-rate normalization for combining multiple on/off-line Japanese character classifiers tested on a large database. In *Proc. 4th Multiple Classifier Systems*, pages 196–205, 2003.
 22. O. Velek, C.-L. Liu, S. Jaeger, and M. Nakagawa. An improved approach to generating realistic Kanji character images from on-line characters and its benefit to off-line recognition performance. In *Proc. 16th Int. Conf. on Pattern Recognition*, pages 588–591, 2002.
 23. A. Vinciarelli and S. Bengio. Writer adaptation techniques in HMM based off-line cursive script recognition. *Pattern Recognition Letters*, 23(8):905–916, 2002.

16 *M. Liwicki and H. Bunke*

24. A. Vinciarelli and M. Perrone. Combining online and offline handwriting recognition. In *Proc. 7th Int. Conf. on Document Analysis and Recognition*, pages 844–848, 2003.
25. A. Waibel, T. Schultz, M. Bett, R. Malkin, I. Rogina, R. Stiefelwagen, and J. Yang. Smart: The smart meeting room task at ISL. In *Proc. IEEE ICASSP*, volume 4, pages 752–755, 2003.
26. M. Wienecke, G.A. Fink, and G. Sagerer. Toward automatic video-based whiteboard reading. *Int. Journal on Document Analysis and Recognition*, 7(2-3):188–200, 2005.
27. R. Zanibbi, D. Blostein, and J.R. Cordy. A survey of table recognition: Models, observations, transformations and inferences. *Int. Journal on Document Analysis and Recognition*, 7(1):1–16, 2004.
28. M. Zimmermann and H. Bunke. Hidden markov model length optimization for handwriting recognition systems. In *Proc. 8th Int. Workshop on Frontiers in Handwriting Recognition*, pages 369–374, 2002.
29. M. Zimmermann and H. Bunke. Optimizing the integration of a statistical language model in HMM-based offline handwritten text recognition. In *Proc. 17th Int. Conf. on Pattern Recognition*, pages 541–544, 2004.



Marcus Liwicki received his M.S. degree in computer science from the Free University of Berlin, Germany, in 2004. Currently he is a Ph.D. student and lecture assistant in the research group of Computer Vision and Artificial Intelligence at the University of Bern, Switzerland.

His research interests include on-line and off-line handwriting recognition and document analysis. He has 9 publications, including a journal paper.



Horst Bunke received his M.S. and Ph.D. degrees in Computer Science from the University of Erlangen, Germany. In 1984, he joined the University of Bern, Switzerland, where he is a professor in the Computer Science Department.

He was Department Chairman from 1992-1996 and Dean of the Faculty of Science from 1997 to 1998, and a member of the Executive Committee of the Faculty of Science from 2001 to 2003. From 1998 to 2000 Horst Bunke was 1st Vice-President of the International Association for Pattern Recognition (IAPR). In 2000 he also was Acting President of this organization.

Horst Bunke is a Fellow of the IAPR, former Editor-in-Charge of the International Journal of Pattern Recognition and Artificial Intelligence, Editor-in-Chief of the journal Electronic Letters of Computer Vision and Image Analysis, Editor-in-Chief of the book series on Machine Perception and Artificial Intelligence by World Scientific Publ. Co., Associate Editor of Acta Cybernetica, the International Journal of Document Analysis and Recognition, and Pattern Analysis and Applications. Horst Bunke held visiting positions at the IBM Los Angeles Scientific Center (1989), the University of Szeged, Hungary (1991), the University of South Florida at Tampa (1991, 1996, 1998-2006), the University of Nevada at Las Vegas (1994), Kagawa University, Takamatsu, Japan (1995), and Curtin University, Perth, Australia (1999), and Australian National University, Canberra (2005). He served as a co-chair of the 4th Int. Conference on Document Analysis and Recognition held in Ulm, Germany, 1997 and as a Track Co-Chair of the 16th and 17th Int. Conference on Pattern Recognition held in Quebec City, Canada and Cambridge, UK in 2002 and 2004, respectively. Also he was chairman of the IAPR TC2 Workshop on Syntactic and Structural Pattern Recognition held in Bern 1992, and was a co-chair of the 7th IAPR Workshop on Document Analysis Systems in Nelson, NZ, 2006. Horst Bunke was on the program and organization committee of many other conferences and served as a referee for numerous journals and scientific organizations. He is on the Scientific Advisory Board of the German Research Center for Artificial Intelligence (DFKI). He has more than 500 publications, including 33 authored, co-authored, edited or co-edited books and special editions of journals.