

# Writer-Dependent Recognition of Handwritten Whiteboard Notes in Smart Meeting Room Environments

Marcus Liwicki, Andreas Schlapbach, and Horst Bunke  
Department of Computer Science, University of Bern,  
Neubrückstr. 10, CH-3012 Bern, Switzerland  
{liwicki, schlpbch, bunke}@iam.unibe.ch  
German Research Center for AI (DFKI GmbH)  
Knowledge Management Department, Kaiserslautern, Germany  
{Marcus.Liwicki}@dfki.de

## Abstract

*In this paper we present a writer-dependent handwriting recognition system based on hidden Markov models (HMMs). This system, which has been developed in the context of research on smart meeting rooms, operates in two stages. First, a Gaussian mixture model (GMM)-based writer identification system developed for smart meeting rooms identifies the person writing on the whiteboard. Then a recognition system adapted to the individual writer is applied. Two different methods for obtaining writer-dependent recognizers are proposed. The first method uses the available writer-specific data to train an individual recognition system for each writer from scratch, while the second method takes a writer-independent recognizer and adapts it with the data from the considered writer. The experiments have been performed on the IAM-OnDB. In the first stage, the writer identification system produces a perfect identification rate. In the second stage, the writer-specific recognition system gets significantly better recognition results, compared to the writer-independent recognizer. The final word recognition rate on the IAM-OnDB-t1 benchmark task is close to 80 %.*

## 1 Introduction

The aim of a smart meeting room is to automate standard tasks usually performed by humans in a meeting [15, 16, 18, 24]. These tasks include, for instance, note taking and summarizing the important topics of a meeting. To accomplish this aim, a smart meeting room is equipped with synchronized recording interfaces for audio, video, and handwritten notes.

The challenges posed in smart meeting room research

are manifold. First, smart meeting rooms need identification systems to authenticate the meeting participants and to assign utterances and handwritten notes to their authors. These are based on speech [14], video [7, 20], and handwriting interfaces [13] or on a combination of those systems [2]. Another aspect is the transcription of the recorded raw data. In order to allow indexing and browsing of the recorded data [25], speech [17], video [4], and handwriting recognition systems [12] need to be developed.

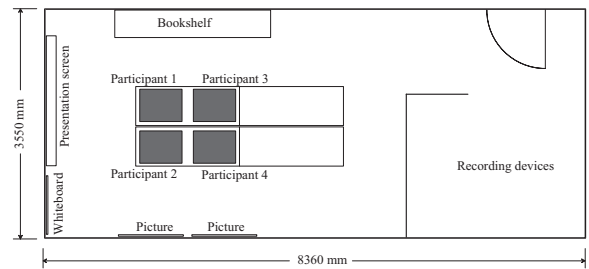
For the latter task, the transcription of handwritten notes, a writer-independent HMM-based system has been proposed recently [12]. However, in the domain of smart meeting rooms, where only a few known persons have access to the whiteboard, it would be reasonable to identify the writer first and then apply a recognition system optimized for this writer.

In this paper a writer-dependent recognition system is proposed. The system operates in two stages. First, a writer identification system for smart meeting room systems recognizes the person writing on the whiteboard. A preliminary version of such a system has been presented in [13]. In the current paper we describe an improved version of this system that uses a better feature set. Then, in the second stage, a recognition system adapted to the individual writer is applied. It is expected that such a two-step approach results in better performance, compared to a writer-independent recognition system, because writer-dependent recognizers have a better knowledge of the writer's individual writing style and have shown a better performance than general recognizers in other application.

Two different methods exist for generating a writer-dependent recognition system. The first method uses the available writer-specific data to train a new recognition system from scratch for each individual writer. The second method takes a writer-independent recognizer and adapts it



**Figure 1. Picture of the IDIAP Smart Meeting Room with the whiteboard to the left of the presentation screen**



**Figure 2. Schematic overview of the IDIAP Smart Meeting Room (top view)**

with the data from the considered writer. While the first method succeeds if enough training data are available, the second method is better if only little writer-specific training data are present [23]. Both methods are investigated in our experiments.

The rest of the paper is structured as follows. Section 2 gives a short overview of the smart meeting room environment for which the handwriting recognition system has been developed. In Sect. 3 we present our writer identification system. The recognition system is introduced in Sect. 4, and the methods for training the individual recognizers are described in Sect. 5. The results of our experiments are presented in Sect. 6. Finally, Sect. 7 draws some conclusions.

## 2 The IDIAP Smart Meeting Room

The writer recognition system described in this paper has been developed for the IDIAP Smart Meeting Room [16]. This meeting room is able to record meetings with up to four participants. It is equipped with multiple cameras, microphones, electronic pens for note-taking, a projector, and an electronic whiteboard. Figure 1 shows a picture of this room, and a schematic overview is presented in Fig. 2.

The whiteboard shown in Figs. 1 and 2 is equipped with the eBeam<sup>1</sup> system, which acquires the text written on the whiteboard in electronic format. A normal pen in a special casing is used to write on the board. The casing sends infrared signals to a triangular receiver mounted in one of the corners of the whiteboard. The acquisition system outputs

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



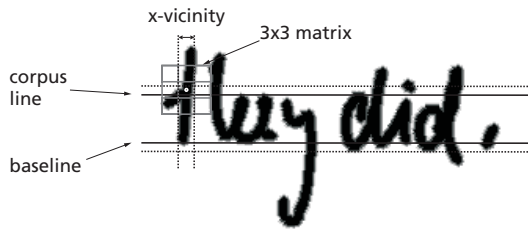
**Figure 3. Recording session**

a sequence of  $(x, y)$ -coordinates representing the location of the pen-tip together with a time stamp for each location. An illustration of the data acquisition process is shown in Fig. 3.

## 3 Writer Identification

We use a GMM-based identification system for writer identification. GMMs are often used in state-of-the-art speaker verification systems [14]. The system is text-independent, i.e., any text can be used to identify the writer. A preliminary version of the identification system has been presented in [13]. Recently, the feature set has been extended, leading to higher identification rates [21]. The novel system is described shortly in this section.

The text written on the whiteboard is encoded as a sequence of time-stamped  $(x, y)$ -coordinates. From this sequence, we extract a sequence of feature vectors and use them to train the identification system. Before feature extraction, some simple preprocessing steps are applied to recover from artifacts, such as spurious points and gaps within



**Figure 4. Features of the off-line matrix**

strokes. In order to preserve writer-specific information, no other normalization operations, such as slant or skew correction, are applied and no resampling of the points is performed. Furthermore, missing points are not interpolated if the distance between two successive points of a stroke exceeds a predefined threshold as this would remove information about the writing speed of a person.

In [13] two feature sets for writer identification have been investigated, stroke-based features and point-based features. In further studies we have developed a better set of point-based features, which clearly outperforms the stroke-based features. The following on-line features are calculated for each point  $p_i$ : the writing direction at  $p_i$ ; the curvature; the relative  $x/y$ -position of the point  $p_i$ ; the speed  $v_i$  and the speed  $v_{i_x}/v_{i_y}$  in  $x/y$ -direction; the acceleration  $a_i$  and the acceleration in  $x/y$ -direction; and the log curvature radius. Additionally, the following features of the on-line vicinity are used: the vicinity aspect; the vicinity slope; the vicinity curliness; and the vicinity linearity. Finally, features of the off-line vicinity are used. They are computed using a two-dimensional matrix  $B = b_{i,j}$  representing the off-line version of the data. For each position  $b_{i,j}$  the number of points on the trajectory of a stroke is stored. This can be seen as a low-resolution image of the handwritten data. The off-line features are the following (see Fig. 4): the number of ascenders above the corpus line and the number of descenders below the baseline in the vicinity of the  $p_i$ ; and the context map, i.e. the number of black points in each region of a  $3 \times 3$  map around  $p_i$ .

Having extracted the features from a text, we use GMMs to model the handwriting of each person of the underlying population. The feature vector sequence  $\mathbf{X} = \mathbf{x}_1, \dots, \mathbf{x}_T$  of a person's handwriting is modeled by a Gaussian mixture density. For a  $D$ -dimensional feature vector  $\mathbf{x}_t$  the mixture density for a specific writer is defined as

$$p(\mathbf{x}_t | \lambda) = \sum_{i=1}^M w_i p_i(\mathbf{x}_t) \quad (1)$$

where the mixture weights  $w_i$  sum up to one. The mixture density is a weighted linear combination of  $M$  uni-modal

Gaussian densities  $p_i(\mathbf{x}_t)$ , each parametrized by a  $D \times 1$  mean vector  $\mu_i$  and a  $D \times D$  covariance matrix  $C_i$ :

$$p_i(\mathbf{x}_t) = \frac{1}{(2\pi)^{D/2} |C_i|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_t - \mu_i)'(C_i)^{-1}(\mathbf{x}_t - \mu_i)\right\}. \quad (2)$$

The parameters of a writer's density model are denoted as  $\lambda = \{w_i, \mu_i, C_i\}$  for all  $i = 1, \dots, M$ . This set of parameters completely describes the model and enables us to concisely model a person's writing on the whiteboard.

The following two-step training procedure is used. In the first step, all training data from all writers are used to train a single, writer independent *universal background model (UBM)*. In the second step, for each writer a writer dependent *writer model* is built by updating the trained parameters in the UBM via adaptation using all the training data from this writer. We derive the hypothesized writer model by adapting the parameters of the UBM using the writer's training data and a form of Bayesian adaptation called *Maximum A Posteriori (MAP)* estimation [19]. For further details see [13].

For identifying the writer of a feature vector sequence  $\mathbf{X} = \mathbf{x}_1, \dots, \mathbf{x}_T$ , the log-likelihood scores of all trained GMMs are taken. The sequence  $\mathbf{X}$  is then assigned to the writer corresponding to the GMM with the highest log-likelihood score.

## 4 Recognition System

The writer-independent text recognition system which serves as a basis for adaptation in the second stage of the overall system has been introduced in [12]. For the purpose of completeness a short summary of this system is provided below.

The text lines on the whiteboard usually have no uniform skew along the whole line and the slant and size of the letters are not the same at the beginning and at the end of a line. This is caused by the fact that people stand, rather than sit, during writing and the arm does not rest on a table. Therefore the text line is split into subparts and the rest of the preprocessing is done for each subpart separately. The individual parts are corrected with respect to their skew. For this purpose we perform a linear regression through all points. For slant normalization, we compute the histogram over all angles between the lines connecting two successive points of the trajectory and the horizontal line [8]. After normalization, delayed strokes, e.g. the crossing of a "t" or the dot of an "i" are removed, using simple heuristics. That is, strokes written in the upper region above already written parts, followed by a pen-movement to the right, are eliminated. Next, we perform an equidistant resampling of the point sequence, i.e. the original sequence of points is

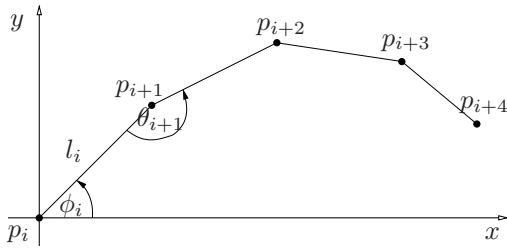


Figure 5. Illustration of point-based features

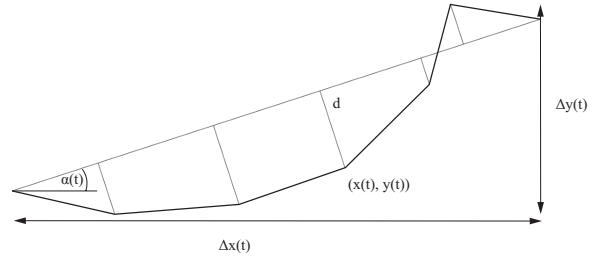


Figure 6. Features of the vicinity

replaced by a sequence of points on the trajectory where all consecutive points have the same distance to each other. The optimal value of the distance has been optimized empirically. This step is needed because different writers write at a different speed. The next important step is the computation of the baseline and the corpus line. For this purpose, we compute two linear regressions through the minima and maxima of the  $y$ -coordinates of the strokes and remove the least fitting points. This correction step is done twice which then results in the estimated baseline (minima) and corpus line (maxima). This results in the three main writing areas. The  $y$ -coordinate of the baseline is then subtracted from all  $y$ -coordinates to make it equal to the  $x$ -axis. As the last preprocessing step, the width of the characters is normalized.

The feature set for recognition contains 25 feature values. Most of them are similar to the features described in Sect. 3. The set of extracted features can be divided into two classes. The first class consists of features extracted for each point  $p_i$  by considering the neighbors with respect to time (see Figs. 5 and 6). The features belonging to this class are the following: the pen-up/pen-down feature which indicates whether the pen-tip has been on the board at the actual position  $p_i$ ; the hat-feature which indicates if a delayed stroke has been removed at the same horizontal position as  $p_i$ ; the speed; the normalized  $x$  and  $y$ -coordinate; the cosine and sine of the writing direction ( $\phi_i$  in Fig. 5); and the cosine and sine of the curvature ( $\theta_i$  in Fig. 5). Again, features of the on-line vicinity are extracted (see Fig. 6): the vicinity aspect, i.e. the aspect of the trajectory in a given on-line vicinity; the vicinity slope, i.e. the cosine and sine of the angle  $\alpha(t)$  of the straight line from the first to the last vicinity point; the vicinity curliness, i.e. the length of the trajectory in the vicinity divided by  $\max(\Delta x(t), \Delta y(t))$ ; and the vicinity linearity, i.e. the average square distance  $d^2$  of each point in the vicinity to the straight line from the first to the last vicinity point. The features of the second class are all computed using a two-dimensional matrix representing the off-line version of the data. The following features are used: the ascenders and descenders, and the nine features extracted from the context map.

An HMM-based recognizer is applied for retrieving the

output label sequence. It uses a linear topology for the character HMMs, and the continuous observation functions are modeled with diagonal Gaussian mixtures. For training, the Baum-Welch algorithm[3] is applied. In the recognition phase, the Viterbi algorithm[5] 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. The number of Gaussian mixtures and the parameters for including a language model are optimized on a validation set.

## 5 Individual Recognizers

Two different methods exist for generating a writer-dependent recognition system. The first is to use the data from the considered writer to train a new recognition system from scratch. We followed this method and generated an individual handwriting recognition system for each writer in the considered population, using the approach described in Sect. 4.

The second method takes a writer-independent recognizer and adapts it with the data from the considered writer. HMM adaptation is a method to adjust the model parameters  $\theta$  of a given background model (the HMMs of the writer-independent recognizer in our case) to the parameters  $\theta_{ad}$  of the adaptation set of observations  $O$  (the data of the individual writer). The aim is to find the vector  $\theta_{ad}$  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  $\theta$  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  $\theta$  is likely to be, Maximum Likelihood Linear Regression (MLLR[10]) 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 is done with Maximum A Posteriori (MAP) estimation, which is also called Bayesian Adaptation. As described in [23], it is feasible to adopt only the means of the Gaussian mixture components  $\mu_{jm}$  (where  $m$  refers to the actual state and  $j$  is the index of the considered mixture in state  $m$ ) of the parameters  $\theta$  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,  $\mu_{jm}$  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 component. After each iteration the values of  $\hat{\mu}_{jm}$  are used in the Gaussian components, 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  $\tau$  weights the influence of the background model on the adaptation data. Whereas  $\tau$  has been set empirically in [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 [23] on adaptation sets of larger size, i.e. about 200 words. This is due to the fact that it adapts each Gaussian component separately. In our experiments we have at least 160 words from each writer available for adaptation.

## 6 Experiments and Results

The experiments have been performed on the IAM-OnDB [11]<sup>2</sup>. This database contains 86,272 word instances from a 11,050 word dictionary written down in 13,040 text lines. The database is divided into four disjoint sets: a training set containing 5,364 lines; a first validation set containing 1,438 lines; a second validation set containing 1,518 lines which can be used, for example, to optimize a language model; and a test set containing 3,859 lines. No writer appears in more than one set. This writer-independent setup serves for training and testing the reference system.

<sup>2</sup><http://www.iam.unibe.ch/~fki/iamondb/>

We performed experiments with a *closed vocabulary* and with an *open vocabulary*. The *closed vocabulary* contains all the words of the test set, while the *open vocabulary* contains the 20,000 most frequent words of three corpora, the LOB corpus [9] (excluding the data used in the IAM-OnDB), the Brown corpus [6], and the Wellington corpus [1]. This setup is similar to the benchmark tasks provided for this database, IAM-OnDB-t1 and IAM-OnDB-t2. For both benchmarks a language model is used. For the *closed vocabulary* it is trained on the LOB Corpus, while it is trained on the three corpora mentioned above for the *open vocabulary*.

For the writer-dependent experiments the test set has been changed as follows. Each writer contributed eight texts, so the data can be separated on the paragraph level. The data is divided into four sets for each writer, containing two paragraphs each. A cross validation is performed on the four sets ( $s_0, \dots, s_3$ ). It is performed in the following way (combinations  $c_0, \dots, c_3$ ). For  $i = 0, \dots, 4$ , sets  $s_{i \oplus 2}$ , and  $s_{i \oplus 3}$  are taken for training the system, set  $s_{i \oplus 1}$  is used as a validation set, i.e., for optimizing the training or adaptation parameters, and set  $s_i$  is used as a test set for measuring the system performance. This cross validation guarantees that each paragraph from the test set is used once for recognition. Therefore the results are comparable to the results of the reference system.

The background model for the writer identification step, as well as the recognition system for the adaptation experiments, are trained on the training set ( $s_{i \oplus 2} \cup s_{i \oplus 3}$ ). Optimization of these models is performed on the validation set ( $s_{i \oplus 1}$ ).

The results of the experiments are reported as word recognition rates and word level accuracies. The accuracy is defined by the following standard formula:

$$acc = 1 - \frac{\#insertions + \#substitutions + \#deletions}{total\_length\_of\_test\_set\_transcription}, \quad (7)$$

where the number of insertions, substitutions and deletions is summed over the whole test set. Note that the accuracy can also have a negative value. This is the case when a large number of insertions has been made, i.e., when the number of insertion errors is larger than the number of correct words in the transcription.

The writer-independent recognition system without optimizing the language model parameters serves as a reference system. To avoid any bias from the optimized language model, only the unoptimized system is taken as a reference.

Tables 1 and 2 show the results of the writer-dependent recognition systems on the *closed vocabulary* and the *open vocabulary*. The first stage always gives a perfect identification rate on the test set of 68 writers. Hence, the correct specialized recognition system is always applied. As it can be seen in Tables 1 and 2, training the writer-dependent recog-

**Table 1. Word recognition accuracy on *closed vocabulary***

System	Rec. (%)	Acc. (%)
Reference	73.1	63.1
Trained from scratch	37.8	0.2
Adapted from reference	<b>75.0</b>	<b>66.7</b>
Optimized language model	<b>77.0</b>	<b>73.2</b>

**Table 2. Word recognition accuracy on *open vocabulary***

System	Rec. (%)	Acc. (%)
Reference	56.2	31.3
Trained from scratch	26.1	-36.8
Adapted from reference	<b>59.7</b>	<b>37.7</b>
Optimized language model	<b>72.6</b>	<b>64.8</b>

nizers from scratch results in much lower recognition rates in all cases. This is mainly due to the low amount of training data. However, adapting the reference system on the writer-specific training data leads to a significant improvement of the accuracy and the recognition rate in all cases (using a standard *z-test* at a significance level of 5%). By optimization of the bigram language model, the recognition rate can further be improved.

## 7 Conclusions

In this paper a writer-dependent recognition system for smart meeting rooms is described. It benefits from the fact that the persons writing on the whiteboard are known to the smart meeting room systems. The proposed system works in two stages. In the first stage, the identity of the writer is determined, and in the second stage, a writer-specific recognition system is applied to recognize the handwritten input.

For the first stage, a language and text-independent system to identify the writer of on-line handwriting is used. New features are extracted from the acquired data to train and test the GMMs. All data are used to train a Universal Background Model (UBM), and client specific models are adapted for each writer. During the identification, a text line of unknown origin is presented to each of the models. Each model returns a log-likelihood score for the given input, and the text line is assigned to the model which produces the highest score.

For the second stage, an HMM-based system is used to train the writer-dependent recognizers. Two different methods for deriving the writer-dependent recognizers have been proposed. The first method uses the available data to

train a new recognition system from scratch. The second recognition system takes a writer-independent recognizer and adapts it with the data from the considered writer. Both methods are investigated in this paper.

Two recognition tasks defined for the IAM-OnDB are used in the experiments. On the test set of each of these tasks, the writer identification system produces a perfect identification rate. Thus, the correct writer-specific recognition system is always applied in the recognition phase.

The recognition experiments show that training a writer-specific recognizer from scratch does not lead to a good performance. However, adapting the system from a well-trained writer-independent recognition system leads to significantly better results. The final word recognition rate on the *closed vocabulary* is already close to 80%, which means that only about one out of five words is not recognized correctly. It can be concluded that selecting writer-specific recognizers by a writer identification system is a useful strategy in the specific domain.

The recognition system has been developed for handwriting data acquired by the eBeam whiteboard system. However, the approach can easily be applied to other on-line handwritten data, such as data acquired by a digitizing tablet or a Tablet PC [22]. For example there could be several optimized recognizers for several writing styles. If a novel user writes on the tablet, his or her handwriting could be assigned to the most similar style.

## Acknowledgments

This work was supported by the Swiss National Science Foundation program “Interactive Multimodal Information Management (IM)2” in the Individual Project “Access and Content Protection”, as part of NCCR. The authors would like to thank Dr. Darren Moore and Maël Guillemot for helping us with technical issues of the IDIAP Smart Meeting Room.

## References

- [1] L. Bauer. *Manual of Information to Accompany the Wellington Corpus of Written New Zealand English*. Department of Linguistics, Victoria University, Wellington, New Zealand, 1993.
- [2] J. Czyz, S. Bengio, C. Marcel, and L. Vandendorpe. Scalability analysis of audio-visual person identity verification. In *Proc. 4th Int. Conf. on Audio- and Video-based Biometric Person Authentication*, pages 752–760, 2003.
- [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] B. Fasel and J. Luetttin. Automatic facial expression analysis: a survey. *Pattern Recognition*, 36(1):259–275, 2003.

- [5] G. D. Forney. The Viterbi algorithm. In *Proc. IEEE*, volume 61, pages 268–278, 1973.
- [6] W. N. Francis and H. Kucera. *Manual of Information to Accompany A Standard Corpus of Present-Day Edited American English, for use with Digital Computers*. Department of Linguistics, Brown University, Providence, USA, 1979.
- [7] M. A. Grudin. On internal representations in face recognition systems. *Pattern Recognition*, 33(7):1161–1177, 2000.
- [8] S. Jäger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: the NPen++ recognizer. *Int. Journal on Document Analysis and Recognition*, 3(3):169–180, 2001.
- [9] S. Johansson. *The tagged LOB Corpus: User's Manual*. Norwegian Computing Centre for the Humanities, Norway, 1986.
- [10] 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.
- [11] 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.
- [12] M. Liwicki and H. Bunke. HMM-based on-line recognition of handwritten whiteboard notes. In *Proc. 10th Int. Workshop on Frontiers in Handwriting Recognition*, pages 595–599, 2006.
- [13] M. Liwicki, A. Schlapbach, H. Bunke, S. Bengio, J. Mariéthoz, and J. Richiardi. Writer identification for smart meeting room systems. In *Proc. 7th IAPR Workshop on Document Analysis Systems*, volume 3872 of *LNCIS*, pages 186–195. Springer, 2006.
- [14] J. Mariéthoz and S. Bengio. A comparative study of adaptation methods for speaker verification. In *Int. Conf. on Spoken Language Processing*, pages 581–584, Denver, CO, USA, September 2002.
- [15] L. McCowan, D. Gatica-Perez, S. Bengio, G. Lathoud, M. Barnard, and D. Zhang. Automatic analysis of multi-modal group actions in meetings. *IEEE TPMAI*, 27(3):305–317, 2005.
- [16] D. Moore. The IDIAP smart meeting room. Technical report, IDIAP-Com, 2002.
- [17] N. Morgan, D. Baron, J. Edwards, D. Ellis, D. Gelbart, A. Janin, T. Pfau, E. Shriberg, and A. Stolcke. The meeting project at ICSI. In *Human Language Technologies Conf.*, pages 246–252, 2001.
- [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] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.
- [20] C. Sanderson and K. K. Paliwal. Fast features for face authentication under illumination direction changes. *Pattern Recognition Letters*, 24(14):2409–2419, 2003.
- [21] A. Schlapbach, M. Liwicki, and H. Bunke. A writer identification system for on-line whiteboard data. 2008. accepted.
- [22] L. Schomaker. From handwriting analysis to pen-computer applications. *IEE Electronics & Communication Engineering Journal*, 10(2):93–102, 1998.
- [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.
- [24] 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.
- [25] P. Wellner, M. Flynn, and M. Guillemot. Browsing recorded meetings with Ferret. In *Machine Learning for Multimodal Interaction*, pages 12–21, 2004.