

# Feature Selection for On-Line 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.** In this paper we describe feature selection experiments for on-line handwriting recognition. We performed a sequential forward search through a 25 elements set of on-line and pseudo-off-line features. In our experiments we obtained interesting results. Using a set of only five features, we achieved a performance that is similar that of the reference system that uses all features. The best subset, which consists of 16 features, could outperform the reference system.

## 1. Introduction

Although the problem of handwriting recognition has been considered for more than 30 years (Plamondon and Srihari, 2000; Vinciarelli, 2002; Bunke, 2003), there are still many open issues, especially in the task of unconstrained handwritten sentence recognition. Handwriting recognition is traditionally divided into on-line and off-line recognition. In on-line recognition a time ordered sequence of coordinates representing the movement of the tip of the pen is captured, while in the off-line mode only the image of the text is available. In this paper we consider an on-line recognition problem, namely the recognition of notes written on a whiteboard. This is a relatively new task. As people stand, rather than sit, during writing and the arm does not rest on a table, handwriting rendered on a whiteboard is different from handwriting produced with a pen on a writing tablet. Despite the additional difficulties, the whiteboard modality is important in several applications, such as the documentation of lectures or meetings. In the particular application underlying this paper we aim at developing a handwriting recognition system to be used in a smart meeting room scenario (Waibel et al., 2003; Moore, 2002). 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.

In Liwicki and Bunke (2006) an on-line recognition system for handwritten whiteboard notes has been presented. This system is based on 25 features gathered from the on-line and off-line vicinity of each sampling point. However, it is an open question whether the extracted features are optimal or near-optimal. Actually, the features may not be independent of each other or be redundant. Some of the features may even have an adversarial effect on the recognition accuracy. In this paper we analyze the set of features and perform a sequential forward search for feature subset selection (Kudo and Sklansky, 2000). In our experiments we found out that a set of five features already produces good results, and a set of 16 features outperforms the reference system that uses all 25 features.

The rest of the paper is organized as follows. Section 2 gives a brief summary of the recognition system. In Section 3 the features are introduced and Section 4 describes the search algorithm. Experiments and results are presented in Section 5 and, finally, Section 6 draws some conclusions and gives an outlook for future work.

## 2. Recognition System

The eBeam interface<sup>1</sup> is used for recording the handwriting. It allows one 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 recognition system described in this paper consists of three main modules: the on-line preprocessing, where noise in the raw data is reduced and the text line is normalized with respect to skew, slant, width and height; the feature extraction, where the sequence of points is transformed into a sequence of feature vectors; and the recognition, where an ASCII transcription of the handwriting is generated.

The recorded on-line data usually contain noisy points and gaps within strokes, which are caused by loss of data. Hence, we apply some noise filtering operations first. The cleaned text data is then automatically divided into lines using some simple heuristics. As the skew often significantly varies within

---

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



Figure 1. Illustration of the recording

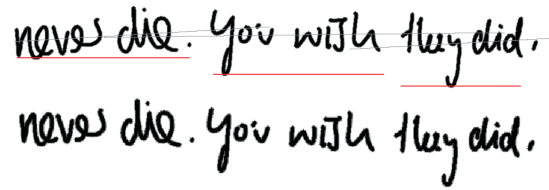


Figure 2. Splitting a text line into subparts and skew correction

the same line, we split lines into subparts of nearly constant skew. An example of splitting is shown in Fig. 2 (upper line). Next the subparts are corrected with respect to their skew, using a linear regression. This process is illustrated in Fig. 2 with the resulting text line shown in the lower part. For slant normalization, we compute the histogram over all angles between the lines connecting two successive points of the trajectory and the horizontal line (Jäger et al., 2001). Subsequently, the histogram is processed to recover the skew angle. After these operations, we remove delayed strokes, e.g. the crossing of a “t” or the dot of an “i”, using simple heuristics. The next important step is the computation of the baseline and the corpus line by computing two linear regression lines through the minima and maxima of the  $y$ -coordinates of the strokes. The baseline is 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 set of extracted features is described in the next section.

After feature extraction an HMM-based recognition system is applied. One HMM is built for each of the 58 characters in the character set, which includes all small and all capital letters together with 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.

### 3. Features

The set of extracted features has been introduced in Liwicki and Bunke (2006). It can be divided into two classes. The first class consists of features extracted for each point  $p_i$  considering the neighbors of  $p_i$  with respect to time. The second class takes the off-line matrix representation of the handwriting into account. For the purpose of completeness, the features are briefly described in this section.

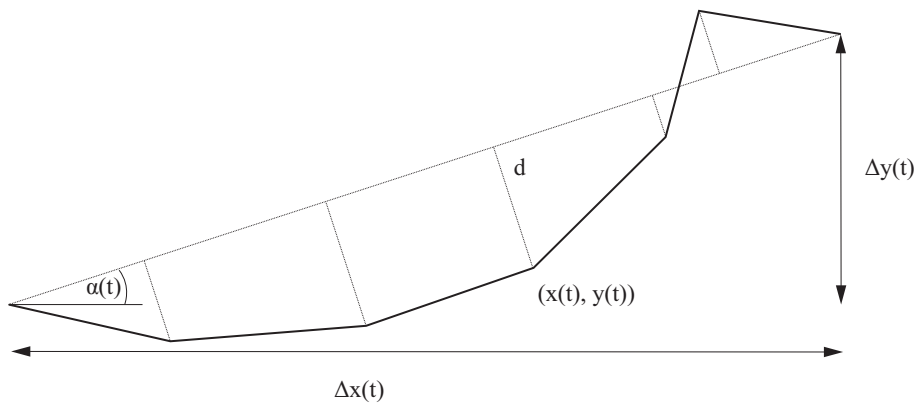
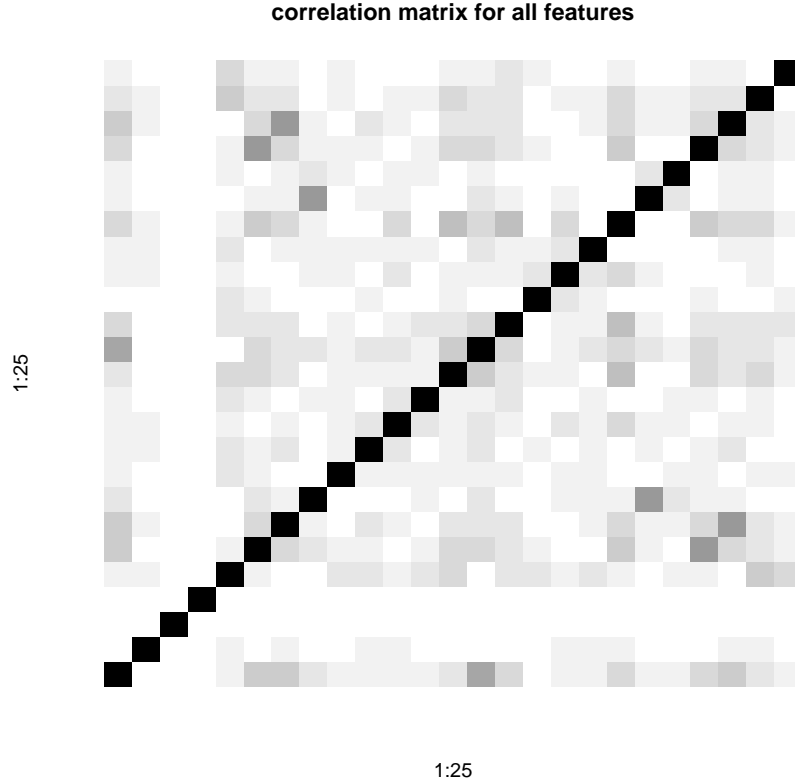


Figure 3. Features of the vicinity



**Figure 4.** Correlation matrix for all features; light colors imply less correlation

The features belonging to the first class are the following<sup>2</sup>: the pen-up/pen-down feature(1) which indicates whether the pen-tip has been on the board at the actual point; the hat-feature(2) which indicates whether a delayed stroke has been removed at the same horizontal position; the speed(3); the normalized  $x$ -coordinate(4); the normalized  $y$ -coordinate(5); the cosine(6) and sine(7) of the writing direction; the cosine(8) and sine(9) of the curvature; the vicinity aspect(19), i.e. the aspect of the trajectory in a given on-line vicinity  $(\Delta y(t) - \Delta x(t)) / (\Delta y(t) + \Delta x(t))$  (see Fig. 3); the vicinity curliness(20), i.e. the length of the trajectory in the vicinity divided by  $\max(\Delta x(t), \Delta y(t))$ ; the vicinity linearity(21), 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; and the vicinity slope, i.e. the cosine(22) and sine(23) of the angle  $\alpha(t)$  of the straight line from the first to the last vicinity point.

All the features of the second class are computed using a two-dimensional matrix representing the off-line version of the data. The following features are used: the ascenders(24) and descenders(25), i.e. the number of points above/below the corpus line with a minimal distance to it and  $x$ -coordinates in the vicinity of the considered point (the distances are set to a predefined fraction of the corpus height); and the context map, where the two-dimensional vicinity of the point is transformed to a  $3 \times 3$  map and the resulting nine values(10-18) are taken as features.

Figure 4 illustrates the correlation matrix of these 25 features. Note that the features are ordered according to their numbers (see above), beginning in the bottom-left corner. In this figure darker values indicate a stronger correlation between the corresponding features. The matrix shows that most of the features have a low correlation.

#### 4. Search Method

In this paper we have applied the sequential forward search (SFS) on a validation set for selecting feature subsets. Given a set of features  $S_1 = \{f_1, \dots, f_n\}$  the algorithm works as follows:

- (1) Start with the feature  $f_s$  that individually performs best on the validation set and put it into the set of best features  $B_1$ ; then set  $S_2 = S_1 \setminus \{f_s\}$ .
- (2) For  $k = 2, \dots, n - 1$  do:
  - For all  $f_i \in S_k$  calculate the performance of  $B_{k-1} \cup \{f_i\}$  on the validation set and rank the features according to their performance.

<sup>2</sup> the numbers in brackets will be used to refer to selected features later in this paper

- Add the best performing feature  $f_i$  to the set of best features  $B_k = B_{k-1} \cup \{f_i\}$  and set  $S_{k+1} = S_k \setminus \{f_i\}$ .

The feature subset  $B_k$  finally selected is the one that performs best among all subsets considered by the algorithm. For a general description of feature selection algorithms we refer to Pudil, Novovičová and Kittler (1994); Kudo and Sklansky (2000).

In Günter and Bunke (2004) it has been pointed out that the number of Gaussians and training iterations have an effect on the recognition results of an HMM recognizer. Often the optimal value increases with the number of features and the amount of training data because more variations are encountered. Since the optimization of this value is very time consuming, we empirically set it for each number of features equal to the optimal values of previous recognition experiments. However, after the optimal feature subset of a given size has been found by SFS, we optimize the number of Gaussians for this set.

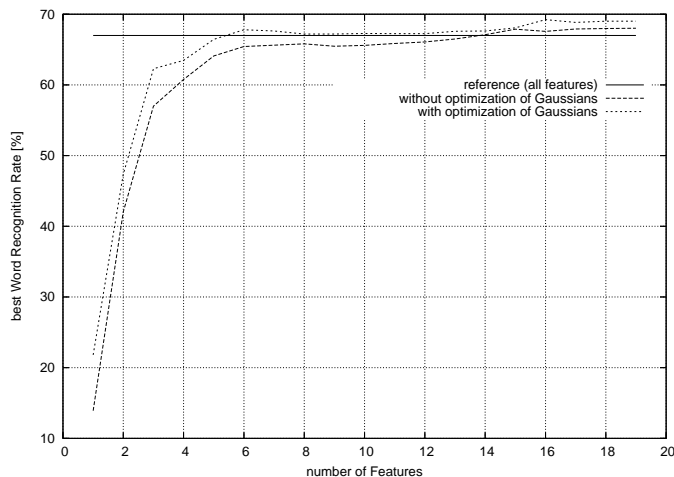


Figure 5. Results of the best combination for each number of features on the validation set

## 5. Experiments and Results

For our experiments we used the IAM-OnDB, a large on-line handwriting database acquired from a whiteboard (Liwicki and Bunke, 2005)<sup>3</sup>. This database contains 86,272 word instances from a 11,050 word dictionary written in 13,040 text lines. Together with this database, there are two sets of benchmark tasks defined. For the experiments described in this paper, we used the sets of the benchmark task with the closed vocabulary IAM-OnDB-t1. There the data is divided into four sets: one set for training; one set for validating the meta parameters of the training; a second validation set which can be used, for example, for optimizing a language model; and an independent test set. No writer appears in more than one set. Thus, a writer independent recognition task is considered. The size of the vocabulary is about 11 K and the recognition rate will always be measured on the word level. In our experiments, we did not include a language model. Thus the second validation set has not been used. To obtain a reference system the recognizer has been trained and tested using all features and 20 Gaussians.

Figure 5 shows the development of the recognition rate during the forward search. As described in the previous section we optimized the number of Gaussians after the best feature combinations have been found. Using a subset of only five feature values, the recognition rate is already close to the recognition rate of the reference system. The highest performance has been reached with 16 features. Table 1 shows the recognition rates of these three systems on the test set. The recognition rate of 73.88% is statistically significantly higher than the recognition rate using all features (significance level of 95%).

A detailed analysis of the results shows that during the first iterations of the search algorithm the ranking of the first five features does not change (see Table 2). From this observation we can conclude that each of the features is quite stable. Also these features give diverse information, which is more important for the recognition than the information provided by other features. The best five features are the cosine of the slope, the normalized  $y$ -position, the density in the center of the  $3 \times 3$ -matrix, the pen-up/down information, and the sine of the curvature.

Other features often do not increase the performance and even sometimes lead to lower recognition scores. These are the cosine of the curvature, the ascenders and descenders, the linearity, the curliness and the aspect of the on-line vicinity. A possible explanation for the fact that the cosine of the curvature is

<sup>3</sup> <http://www.iam.unibe.ch/~fki/iamondb/>

**Table 1.** Performance of selected subsets on the test set

Subset	Recognition rate in %
Reference system	73.13
5 best features	71.83
Best subset (16)	<b>73.88</b>

**Table 2.** Ranking of features in step two for the first combinations during validation

Iteration $k$	Best subset $B_k$	Ranking of remaining features
1	{6}	5,14,1,9,23,3,...
2	{6,5}	14,1,9,23,3,...
3	{6,5,14}	1,9,3,...
4	{6,5,14,1}	9,18,17,...

not a good choice while the sine of the curvature is one of the best features is the following. While the sine clearly distinguishes between a right turn and a left turn, the cosine does not change significantly for small angles (which are the usual case in the handwriting). Thus it does not provide additional information for the recognizer.

## 6. Conclusions and Future Work

In this paper feature selection experiments for the recognition of handwritten whiteboard notes have been presented. During the sequential forward search a best performing subset of 16 features has been found. This subset outperforms the reference system which uses all 25 features.

An interesting outcome for the handwriting community is that a subset of five features produces already good results. These five features are not highly correlated and have been the top choices during the first iterations of the forward search. Because of the small number of features we get a more efficient classifier with respect to both computation time and memory requirements.

In future we plan to continue the feature selection experiments using other strategies, such as Sequential Backward Search (SBS), Sequential Floating Forward Search (SFFS), and Sequential Floating Backward Search (SFBS) (Pudil, Novovičová and Kittler, 1994).

## Acknowledgements

This work was supported by the Swiss National Science Foundation program “Interactive Multimodal Information Management (IM)<sup>2</sup>” in the Individual Project “Visual/Video Processing”, as part of NCCR.

## References

- Bunke, Horst. 2003. Recognition of Cursive Roman Handwriting – Past Present and Future. In *Proc. 7th Int. Conf. on Document Analysis and Recognition*. Vol. 1 pp. 448–459.
- 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.
- 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.
- Jäger, S., S. Manke, J. Reichert and A. Waibel. 2001. “Online handwriting recognition: the NPen++ recognizer.” *Int. Journal on Document Analysis and Recognition* 3(3):169–180.
- Kudo, Mineichi and Jack Sklansky. 2000. “Comparison of algorithms that select features for pattern classifiers.” *Pattern Recognition* 33(1):25–41.
- Liwicki, M. and H. Bunke. 2005. 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*. Vol. 2 pp. 956–961.
- Liwicki, M. and H. Bunke. 2006. HMM-Based On-Line Recognition of Handwritten Whiteboard Notes. In *Proc. 10th Int. Workshop on Frontiers in Handwriting Recognition*. pp. 595–599.
- Moore, D. 2002. The IDIAP Smart Meeting Room. Technical report IDIAP-Com.
- Plamondon, Rjean and Sargur N. Srihari. 2000. “On-Line and Off-Line Handwriting Recognition: a Comprehensive Survey.” *IEEE Trans. Pattern Analysis and Machine Intelligence* 22(1):63–84.
- Pudil, P., J. Novovičová and J. Kittler. 1994. “Floating search methods in feature selection.” *Pattern Recognition Letters* 15(11):1119–1125.
- Vinciarelli, A. 2002. “A survey on Off-Line Cursive Script Recognition.” *Pattern Recognition* 35(7):1433–1446.
- Waibel, A., T. Schultz, M. Bett, R. Malkin, I. Rogina, R. Stiefelhagen and J. Yang. 2003. SMaRT: the Smart Meeting Room Task at ISL. In *Proc. IEEE ICASSP*. Vol. 4 pp. 752–755.