Machine Vision and Applications (2011) 22:39–51
DOI 10.1007/s00138-009-0208-9

ORIGINAL PAPER

Combining diverse systems for handwritten text line recognition

Marcus Liwicki · Horst Bunke · James A. Pittman · Stefan Knerr

Received: 19 March 2008 / Revised: 28 October 2008 / Accepted: 15 June 2009 / Published online: 23 July 2009

Abstract In this paper, we present a recognition system for on-line handwritten texts acquired from a whiteboard. The system is based on the combination of several individual classifiers of diverse nature. Recognizers based on different architectures (hidden Markov models and bidirectional long short-term memory networks) and on different sets of features (extracted from on-line and off-line data) are used in the combination. In order to increase the diversity of the underlying classifiers and fully exploit the current state-of-the-art in cursive handwriting recognition, commercial recognition systems have been included in the combined system, leading to a final word level accuracy of 86.16%. This value is significantly higher than the performance of the best individual classifier (81.26%).

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 on-line and off-line recognition [4,23,34,37,39]. In this paper, we consider a relatively new task, which is the recognition of text written on a whiteboard. Note that a similar task has been considered in [7,30]. However, while in [7,30] a video camera was used to capture the handwriting, we use the eBeam interface which is based on infrared sensing. This system is easier to use than a video camera and it is less vulnerable to artifacts arising from poor lighting conditions, self-occlusion and low image resolution. This task is relevant in a number of novel applications, for example, computer-assisted learning [17] and smart meeting rooms [29,42]. While the electronic whiteboard has already been a research topic in [1,11], the task of recognizing whiteboard notes written in Roman script is relatively new. Recently, on-line [25] and off-line recognition systems [27] have been proposed by the authors of this paper.

Having on-line and off-line recognition systems available, it may be beneficial to combine both systems. From such a combination, an improved recognition performance can be expected [38,40]. A general overview and an introduction to the field of multiple classifier systems (MCS) is given in [22]. Several combination methods for character, numeral, and word recognition have been proposed in the literature [16,40,44]. However, little work has been reported on the combination of classifiers for general handwritten text line and sentence recognition. The combination of the outputs of multiple handwritten text line and sentence recognizers differs from standard multiple classifier combination, because the output of a text line recognizer is a sequence of word classes rather than just one single word class, and the number of words may differ among several recognizers. Therefore, an additional alignment procedure is needed. In the work

M. Liwicki (✉) · H. Bunke
Institute of Computer Science and Applied Mathematics, University of Bern, Neubrückstrasse 10, 3012 Bern, Switzerland
e-mail: liwicki@iam.unibe.ch

H. Bunke
e-mail: bunke@iam.unibe.ch

J. A. Pittman
Microsoft, Redmond, USA
e-mail: Jay.Pittman@microsoft.com

S. Knerr
Vision Objects, Nantes Cedex 3, France
e-mail: stefan.knerr@visionobjects.com

presented in this paper, the recognizer output voting error reduction (ROVER) [8] framework has been applied. To combine output of several individual recognizers, the word sequences are incrementally aligned using a standard string matching algorithm. Then a voting strategy is applied to get the final result.

In this paper, the combination of different systems derived from recognizers introduced in previous work by the authors [25,27,28] is described. A hidden Markov model (HMM)-based recognizer and a recognizer based on bidirectional long short-term memory networks (BLSTM) are used as the base recognizers of the multiple classifier system. Both base recognizers are trained on three different feature sets derived either from the on-line data directly or from off-line images synthesized from the on-line handwriting. To the best of the authors' knowledge, these are the first attempts in the field of handwritten text line recognition combining systems based on off-line and on-line features (However, we are aware of works combining on-line and off-line features for task different from complete text line recognition). Furthermore, commercial recognition systems from Microsoft© and Vision Objects© have been included in the combination experiments. Since these recognizers are based on different features and classification methods, it can be expected that the performance increases, although the individual recognition rates of these commercial recognizers are lower than those of the neural network based approach. Note that early work on combining on-line and off-line HMM-based systems has already been published in [26]. However, the current paper goes clearly beyond the scope of [26], because different recognition architectures are combined in this paper and furthermore, commercial recognizers are included in the ensemble.

This paper is organized as follows. A description of all recognition systems used in the combination experiments is given in Sect. 2. Next, the methodology for combining the different recognizers is introduced in Sect. 3. Section 4 presents the experimental results, and finally, Sect. 5 draws some conclusions and gives an outlook to future work.

2 Data, features, and recognition systems

The multiple classifier system described in this paper incorporates several individual classifiers. Besides recognizers based on HMMs and neural networks, the Microsoft© handwriting recognition engine [33] and the Vision Objects© recognizer [20,21] have been included in the combination.

This section gives an overview of all recognition systems used in the work described in this paper. First, the handwritten data are presented in Sect. 2.1. Next, the normalization and feature extraction procedures are summarized in Sect. 2.2. Then, the HMM and BLSTM recognizers are described in Sects. 2.3 and 2.4. Section 2.5 presents the Microsoft© handwriting recognition engine, and the Vision Objects© recognizer is shortly described in Sect. 2.6.

2.1 Handwritten data

The eBeam interface 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 positional resolution is ± 1.5 mm, and the sampling rate of the recordings varies from 30 to 70 samples per second. The data returned by the acquisition device are in xml-format. An illustration is shown in Fig. 1.

The recorded on-line data usually contain noisy points and gaps within strokes. Thus, we first apply some on-line preprocessing operations to recover from these artifacts. These operations are described in [27]. The cleaned text data are then automatically divided into lines using some simple heuristics.

2.2 Normalization and feature extraction

Both the HMM and BLSTM recognition systems have been trained and tested on three different feature sets each. The first set of features is based on off-line images generated from the on-line data. To generate the off-line images, consecutive points have been connected and the resulting lines have been dilated. Then each text line is 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

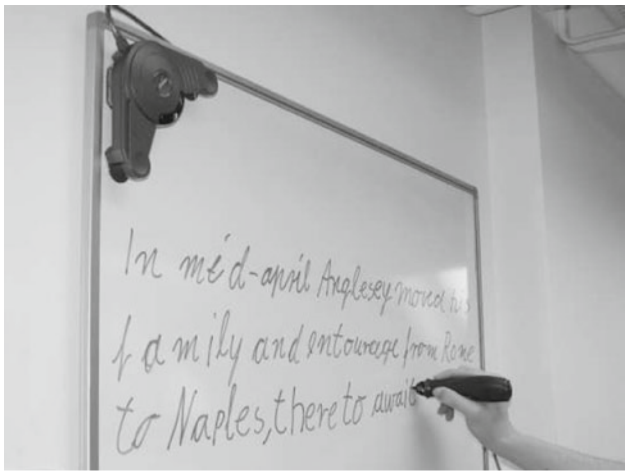

In mid-april Anglesey moved his
family and entourage from Rome
to Naples, there to await

Fig. 1 Illustration of the recording (note the eBeam system at the left upper corner of the whiteboard)

lower baselines), the ascender part (located above the upper baseline), and the descender part (below the lower baseline) are vertically scaled to a predefined size each. Writing width normalization is performed by a horizontal scaling operation. The purpose of this operation 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. For more details about the features and preprocessing refer to [27].

To get the second set of features, the individual text lines are first corrected with respect to their skew using 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 [18]. After normalization, delayed strokes, e.g., the crossing of a "t" or the dot of an "i" are removed, using simple heuristics. Next, we perform an equidistant resampling of the point sequence, i.e., the original sequence of points is replaced by a sequence of points on the trajectory where all consecutive points have the same distance to each other. This step is needed because different writers write at a different speed. The distance is set to a fraction of the corpus height. The next important step is the computation of the baseline and the corpus line. The baseline is subtracted from all y-coordinates to make it equal to the x-axis and the height of the three main writing areas is normalized. As the last preprocessing step, the width of the characters is normalized. The set of extracted features can be divided into two classes. The first class consists of features extracted for each point considering its neighbors with respect to time. The features of the second class are all computed using a two-dimensional matrix representing the off-line version of the data. For more details refer to [25].

The third set of features is basically extracted in the same manner as the second one up to two differences. Before applying the normalization operations, an additional step is introduced, which is important for the whiteboard data. The text lines on a whiteboard usually have no uniform skew along the whole line and the slant and size of the letters is 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. Splitting is done at gaps that are larger than the mean gap size. Also the size of both subparts has to be greater than a predefined threshold. The slant correction is supplemented with the following method. We weight the histogram values with a Gaussian with its mean at the vertical angle, and the variance empirically set. This is beneficial because some words are not properly corrected if a single long straight line is drawn in horizontal direction, which results in a large histogram value. We also smooth each histogram entry with its direct neighbors using the window (0.25, 0.5, 0.25), because in some cases the correct slant is at the border of two angle intervals and a single peak at another interval may be slightly higher. This single peak will become smaller after smoothing.

2.3 Hidden Markov models

The first recognizer is based on hidden Markov models (HMMs). An HMM is built for each of the 82 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. In the emitting states, the observation probability distributions are estimated by mixtures of Gaussian components. In other words, continuous HMMs are used. The number of Gaussian mixtures is optimized on a validation set as suggested in [13]. The character models are concatenated to represent words and sequences of words. For training, the Baum-Welch algorithm [5] is applied. In the recognition phase, the Viterbi algorithm [9] is used to find the most probable word sequence. The output of the recognizer is a sequence of words.

2.4 Neural networks

Recurrent neural networks (RNNs) are able to access a wide range of context when transcribing letters, words or word sequences [35]. However, whereas standard RNNs make use of previous context only, bidirectional RNNs (BRNNs) are able to incorporate context on both sides of every position in the input sequence [36]. This is useful in handwriting recognition since it is often necessary to look at both the context to the right and to the left of a given letter in order to identify it. Figure 2 shows a comparison of a standard RNN and a BRNN at two different points in time. In the BRNN there exist two hidden layers, i.e., one layer for each direction, and

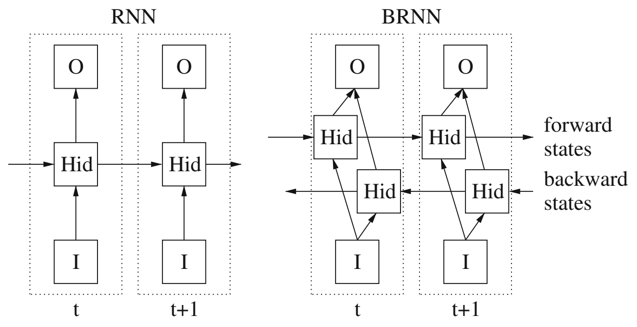


Fig. 2 Recurrent neural network and bidirectional recurrent neural network unfolded in time

the input/output neurons (marked with I/O) are connected to both.

Long Short-Term Memory (LSTM [15]) is an RNN architecture specifically designed to bridge long time delays between relevant input and target events, making it suitable for problems where long-range context is required to disambiguate individual labels as in handwriting recognition. An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks. Each memory block contains one or more recurrently connected memory cells and three multiplicative units, called the input, output, and forget gates that provide write, read, and reset operations on the cells.

The recently introduced connectionist temporal classification (CTC) [6] is an RNN objective function designed for labeling whole sequences at once. It uses the network to define a probability distribution over a fixed set of labels plus an additional "blank", or "no label" unit. It then interprets the sequence of network outputs as a probability distribution over all possible transcriptions for a given input sequence, and trains the network by maximizing the log probabilities of the correct transcriptions on the training set. In our experiments, we use a BRNNs with LSTM network cells and use a CTC output layer to transcribe samples of handwritten text. For more details of this recognizer, we refer to [28].

2.5 Microsoft© handwriting recognition engine

The Microsoft© Tablet PC Software Development Kit (SDK) provides an interface for ink-enabled applications for the tablet PC.[1] It includes an application programming interface (API) to the Microsoft© handwriting recognition engine (HWR), which exists on any computer running Microsoft Windows XP Tablet Edition©[33].

To support a wide range of writing styles, a large neural network was trained on a very large data set. The training set contains ink samples from thousands of people with a diverse range of writing styles. An illustration of how the recognizer works is given in Fig. 3 [33]. First, a stroke is cut into separate segments whenever it moves downward in y-dimension and then reverses its direction back upward. Then a set of features is extracted for each individual segment. Generally, the measurements for the features are based on the direction and curvature of the ink trace, along with various measurements of size. Finally, to accomplish the recognition of connected letters in cursive script, a time-delay neural network (TDNN) with a window size of five is used. The top part of Fig. 3 shows how the electronic ink is processed and fed through the TDNN. The TDNN outputs consist of a matrix, where each column represents one ink segment and each row

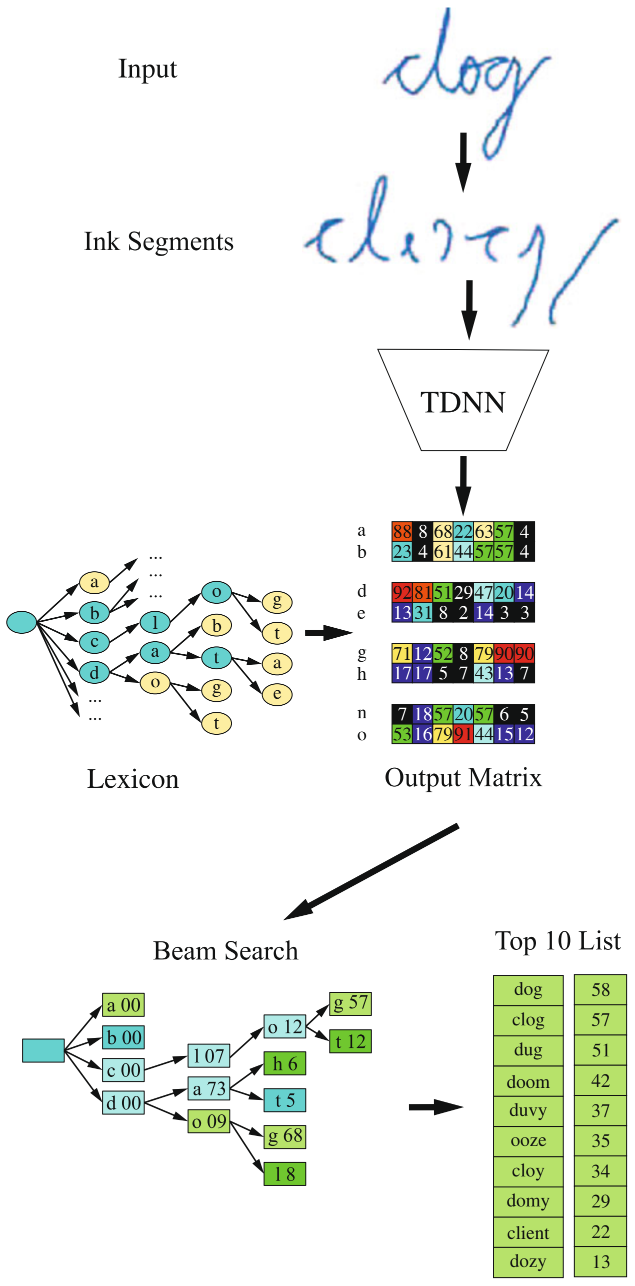


Fig. 3 Illustration of the recognition architecture of the Microsoft© HWR

represents one digit or character. The outputs are estimates of probability for each possible digit or character.

The TDNN is further supported by integrating a language model in form of a lexicon, organized in the form of a trie (see left part of Fig. 3). The nodes of the trie contain a Boolean flag indicating whether the corresponding letter is the end of a word. Optionally, some of these end-of-word nodes also contain a word unigram probability or other information. The lexicon is combined with the output of the TDNN by means of a beam search, an approach that most commercial speech-

[1] The Microsoft Windows XP©X Tablet PC Edition SDK is available for download at http://www.microsoft.com/windowsxp/tabletpc/default.mspx.

recognition systems use [14]. The bottom part of Fig. 3 illustrates how the beam search moves through the TDNN output matrix, column by column, from left to right. At each column, it consults the lexicon on the left to find out what letters might be next and looks up the probability estimates for those letters in the matrix. It builds a copy of the lexicon trie (shown on the right-hand side in Fig. 3), with the scores stored in the nodes of this copy trie. At any one time, the scores in that trie are the cumulative scores of each character of the current column, and all characters up to that character, for each possible word in the lexicon. When a parent node produces a child node, the score in the child node is computed taking the current score in the parent node into account. After the child node has been generated, parent and child operate independently. They may even compete, because it is possible that the column (segment) still represents part of the parent letter, or it is now part of the child letter. Once the last column in the matrix is completed, the trie contains the final hypotheses about what the word might be. At this point, it is simply a matter of treversing the trie to find the *n* best-scoring words.

The communication process with the Microsoft© HWR works as follows. The client application sends on-line stroke information to the recognition engine on the tablet PC where the recognition is performed. The recognition engine answers with several recognition alternatives and their confidences. However, there are only three confidence values available (the number in brackets denotes the actual confidence value of the recognizer):

– *strong* (0): only about 25% of the text lines in the test set of our experiments are recognized with a strong confidence
– *intermediate* (1): this confidence is given for less then 3% of these text lines only
– *poor* (2): in most cases a poor confidence is assigned

For handwritten text lines the alternatives represent different transcriptions of a given input. The confidence value assigned to a transcription represents the lowest confidence level of a recognized segment found in the input. Thus, a strong confidence is rarely given since it is not likely for a recognizer to be confident for a whole text line.

The Microsoft© HWR allows a user to define a vocabulary and a character set. In this paper, the same vocabulary is used as for all recognizers. However, the Microsoft© HWR is not confined to recognizing only words from the given vocabulary. Therefore, often out of vocabulary words (OOVs) occur in the recognition alternatives.

2.6 Vision objects© recognizer

The MyScript recognizer from Vision Objects© is an on-line recognizer that processes digital ink and supports a large spectrum of languages and functionalities for applications that vary from Forms Processing and Note Taking to Mobile Data Capture. Application programming interfaces exist for all major operating systems on the market, both for PC/Server architectures (Linux, Windows, Mac OS) and mobile platforms (Embedded Linux, Symbian, Windows CE/Mobile, and many others).[2]

The overall recognition system is built on the principles presented in [20]. Some of the more important concepts are

– use of a modular and hierarchical recognition system,
– use of soft decisions (often probabilistic) and deferred decisions by means of considering concurrent hypotheses in the decision paths,
– use of complementary information at all stages of the recognition process, and
– use of global optimization criteria, making sure that the recognizer is trained in order to perform optimally on all levels.

The recognizer has been trained on many millions of writing samples that have been collected from native writers in the target countries. The processing chain of the Vision Objects recognizer starts out with some of the usual preprocessing operations, such as text line extraction and slant correction. Then, the on-line handwriting is pre-segmented into strokes and sub-strokes. Here, the general idea is to over-segment the signal and let the recognizer decide later on where the boundaries between characters and words are. This is followed by feature extraction stages, where different sets of features are computed. These feature sets use a combination of on-line and off-line information. The feature sets are processed by a set of character classifiers, which use Neural Networks and other pattern recognition paradigms. All the information accumulated in the various processing steps is then processed by dynamic programming on the word and sentence levels in order to generate character, word, and sentence level candidates with corresponding confidence scores. A global discriminant training scheme on the word level with automatic learning of all classifier parameters and meta-parameters of the recognizer, in the general spirit of what has been described in [21], is employed for the overall training of the recognizer.

The Vision Objects recognizer uses a state-of-the-art statistical language model that incorporates lexical, grammatical, and semantic information. This model is partially described in [31]. By means of the language model, the recognizer uses context information on all recognition levels, from the character to the word, and to the sentence level. The employed language model also supports the recognition of terms that are not explicitly covered by the lexicons (in this paper a lexicon for British English), albeit the recognizer has

[2] The MyScript Builder SDK©X is available for purchase at http://www.visionobjects.com/.

a tendency to convert handwriting into text that is covered by the lexicons.

The communication with the client is similar to the communication of the Microsoft© recognizer. The client sends the stroke information to the recognition engine and gets a recognition result as an answer. This answer is divided into several segments. Each segment contains alternatives for the word at the actual position. The candidates come together with a normalized recognition score. This allows the Vision Objects©recognizer to be used for advanced combination strategies where the recognizer's confidence is exploited.

3 Combination methodology

Multiple classifier systems have been successfully used to improve the recognition performance in difficult tasks [22]. There are two issues in multiple classifier systems research. First, individual classifiers that exhibit a high degree of diversity need to be constructed. The classifiers used in this work have been described in the previous section. As theses classifiers are based on different features and classification paradigms, and have largely been developed independently of each other, it can be expected that their diversity is quite high. The second issue to be addressed in a multiple classifier system is classifier combination. A large number of combination schemes have been proposed [22]. However, these schemes are restricted in the sense that they only address the classification of single objects. By contrast, in the application considered in this paper, sequences of objects, i.e., whole text lines, need to be classified. This means that the task of the recognizer is not only word recognition, but also the segmentation of a line of handwritten text into individual words. Because of segmentation errors, the word sequences output by the individual recognizers may consist of a different number of words. Therefore, an alignment procedure is needed as the first step in classifier combination. For aligning the outputs of the individual classifiers, the ROVER combination strategy has been chosen. This strategy consists of two stages. Because the recognizers output word sequences for whole text lines and because there may be a different number of words in each sequence, the output sequences are aligned into a *word transition network* (WTN) first. A voting strategy is then applied to select the best scoring word at each location for the final transcription. The alignment process is described in Sect. 3.1, and several voting strategies are proposed in Sect. 3.2.

3.1 Alignment

Finding the optimal alignment for n sequences is NP-complete [43]. Thus, an approximate solution for the alignment problem has been chosen. This solution aligns the multiple sequences incrementally by building WTNs. At the beginning, the first and the second word sequence are aligned in a single WTN, using the standard string matching algorithm described in [41]. The resulting WTN is aligned with the next word sequence giving a new WTN, which is then aligned with the next recognition result, and so on. This method does not guarantee an optimal solution, but in practice the suboptimal solution usually provides an alignment of sufficiently high accuracy. An example alignment of the output of three recognizers (denoted by W_1, W_2, and W_3) is shown in Fig. 4. The columns in the WTNs denote sets of corresponding words, called *correspondence sets* in the following. Those correspondence sets are identified by the alignment process. Note that the symbol ϵ marks *null*-transitions, i.e., transitions where an empty string is chosen as an alternative.

Figure 4 contains examples for all possible correspondence set categories that occur during string alignment. The four categories are

– Correct: the word transcriptions are the same ("mid-april" in $\mathbf{WTN_1}$)
– Insertion: an additional word occurs in the new word sequence, resulting in an additional column with *null*-transitions ("a" in $\mathbf{WTN_2}$)
– Deletion: fewer words occur in the new word sequence, and consequently a *null*-transition is inserted ("say" in $\mathbf{WTN_1}$)

In mid-april Anglesey

W_1: In mid-april Angle say

W_2: It mid-april Anglesey

W_3: I a mid-April Anglesey

$$\mathbf{WTN_1 = W_1 + W_2} :$$

In	mid-april	Angle	say
It	mid-april	Anglesey	ϵ

$$\mathbf{WTN_2 = WTN_1 + W_3} :$$

In	ϵ	mid-april	Angle	say
It	ϵ	mid-april	Anglesey	ϵ
I	a	mid-April	Anglesey	ϵ

Fig. 4 Example of iteratively aligning multiple recognition results

– Substitution: the word transcriptions at the same position differ (“Angle”/“Anglesey” and “In”/“It” in $\mathbf{WTN_1}$)

3.2 Voting strategies

After alignment, a voting module extracts the best scoring word sequence from the WTN. One possibility is to take only the number of occurrences of a word w for making a decision. In the case of ties, the output of the best performing system on the validation set is taken. In the example of Fig. 4, assuming that W_1 has a higher recognition accuracy than W_2 and W_3, the output will be “In” from W_1, which yields the final output “In mid-april Anglesey”. Note that this is the correct transcription, which is not present in the recognition result of any single recognizer.

In addition to the frequency of occurrence, the confidence of the recognizers can be used as a voting strategy. The trade off between the frequency of occurrence and the confidence score is weighted with a parameter $\alpha \in [0, 1]$. Let c be a correspondence set of the WTN containing n word classes $w_1, \ldots, w_n$. The number of occurrences of each word class w_i is denoted by N_i. Then the score $S(w_i)$ of a word w_i is

$$S(w_i) = \alpha \cdot \frac{N_i}{\sum_{j=1}^{n} N_j} + (1 - \alpha) * C(w_i), \tag{1}$$

where $C(w_i)$ is the combined confidence score of word class w_i. There exist several methods to calculate $C(w_i)$. First, the average of all confidence scores of the occurrences of word class w_i can be used. Second, the maximum of these confidence scores may be chosen. Setting $\alpha = 1$ results in a voting procedure that takes only the number of occurrences into account, while $\alpha = 0$ corresponds to the case where only the confidence counts.

Another parameter is the confidence score of the *null*-transition $C(\epsilon)$. It determines how often the *null*-transition is taken instead of another word class. If *null*-transitions are taken rather seldomly, i.e., if $C(\epsilon)$ is very low, the output transcription tends to be longer and more insertion errors occur. On the contrary, if $C(\epsilon)$ is too large, more deletions occur. The two parameters α and $C(\epsilon)$ are optimized on a validation set in this paper.

4 Experiments and results

All experiments have been performed on the IAM-OnDB [24], a large on-line handwriting database acquired from a whiteboard.[3] It consists of 13,040 written lines, containing 86,272 instances of 11,050 distinct words. For the alignment and construction of the WTN, the Rover tool from the National Institute of Standards and Technology (NIST) implementing the ROVER framework [8] has been used.[4]

Two benchmark tasks have been defined for the IAM-OnDB. In this paper, the IAM-OnDB-t2 benchmark task has been used for the experiments. In this task, the database is divided into four predefined 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. Thus, a writer-independent recognition task is considered.

The task for all the experiments is to transcribe all the text lines in the test set, given a predefined dictionary. The measure of performance is the *word accuracy*:

$$100 * \left(1 - \frac{\text{insertions} + \text{substitutions} + \text{deletions}}{\text{total length of test set transcriptions}}\right)$$

where the number of word insertions, substitutions, and deletions is summed over the whole test set. The IAM-OnDB-t2 benchmark task is based on an open vocabulary that contains the 20, 000 most frequent words out of three corpora (LOB Corpus [19], Brown Corpus [10], Wellington Corpus [2]). Note that there exist words, called out of vocabulary words (OOV), which are in the transcription of the test set but not in the lexicon. This limits the recognition rate of those recognizers that are only able to recognize words included in the lexicon. On the test set the OOV rate is 5.6%, resulting in an upper limit of 94.4% for the CTC and HMM classifiers.

All systems described in Sect. 2 have been used in the experiments. The parameters for recognition and combination have been optimized on the first validation set. For the recognizers evaluated in this section, a bigram language model has been included. The language model weighting factor and the word insertion penalty are optimized on the validation set. Language model training is based on three different corpora (LOB Corpus, Brown Corpus, Wellington Corpus).

The rest of this section is organized as follows. Section 4.1 reports on the recognition results of the individual recognizers. The optimization of various system parameters on the validation set is described in Sect. 4.2, and the recognition results on the test set are presented in Sect. 4.3.

4.1 Individual recognition results

In addition to the Microsoft© and Vision Objects© recognizers, a total of six classification systems are used in the combination. Three of these systems are based on HMMs while the rest use neural networks. Since neural networks are

[3] The IAM-OnDB is publicly available for download at http://www.iam.unibe.ch/~fki/iamOnDB.

[4] The Rover tool is part of the NIST Scoring Toolkit available for download at ftp://jaguar.ncsl.nist.gov/current_docs/sctk/doc/sctk.htm.

Table 1 Recognizers used for the combination experiments and their performance on the test set

System		#	Accuracy (%) (average for CTC) (%)
CTC	Off-line	10	76.43 (73.6)
	On-line	10	81.26 (79.8)
	Extended on-line	10	81.07 (80.1)
HMM	Off-line	1	57.3
	On-line	1	63.2
	Extended on-line	1	63.9
Microsoft©		1	71.3
Vision Objects©		1	79.2

randomly initialized, it is advisable to consider more than one neural network for each feature set. Thus, a total of ten neural networks have been trained with each of the three feature sets described in Sect. 2.2, resulting in 30 individual neural network based recognizers.

A summary of the individual recognition systems and their accuracies is given in Table 1. Altogether, 35 classifiers are involved in the experiments: 30 randomly initialized CTC networks, three HMM-based classifiers, the Vision Objects© recognizer, and the Microsoft© recognizer. The different feature sets are denoted by "off-line" for off-line features, "on-line" for conventional on-line features and preprocessing, and "extended on-line" for on-line features extracted after applying additional preprocessing steps (see Sect. 2).

While the highest accuracy is obtained by an "on-line" CTC (81.26%), the average accuracy of the "extended on-line" CTC (80.1%) is higher than that of the "on-line" CTC (79.8%). The off-line CTC systems perform with an average accuracy of 73.6%. Note that the standard deviation of the CTC systems is less then 0.5% for each of the three feature sets. The HMM-based recognizer has the lowest accuracy. The recognition rates of the "off-line" and "on-line" system are significantly lower than the "extended on-line" system. The Vision Objects© recognizer is the best commercial recognizer on the IAM-OnDB-t2 benchmark task. Its accuracy of 79.2% on the test set is significantly higher than that of the Microsoft© recognizer which performs at 71.3%.

4.2 Optimization of the MCS on the validation set

Several parameters of the MCS have to be optimized during validation. First, there are the parameters for voting described in Sect. 3. Then, the order of the classifiers for the alignment has to be chosen. Finally, it has to be decided which classifiers are actually to be included in the ensemble. This section will describe how the optimal values of these parameters are found using the first validation set of the IAM-OnDB-t2 benchmark task.

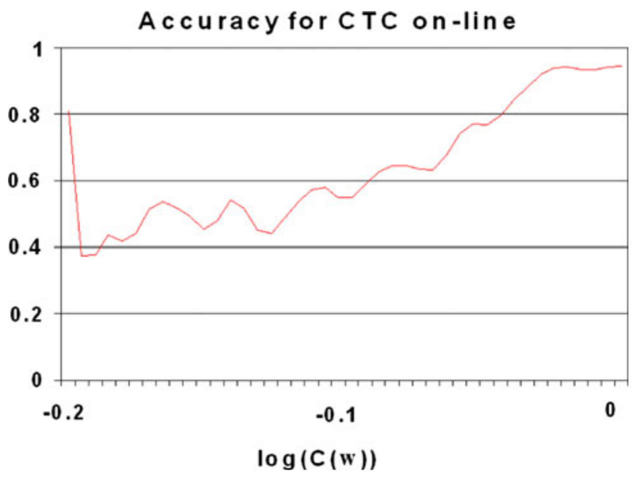


Fig. 5 Accuracy on the validation set for several confidence intervals

4.2.1 Voting

We have used three strategies for the decision which word in a correspondence set is chosen as the final output. First, only the frequency of occurrence is used. In the second and third strategies, confidence scores have been used. Under these strategies the outputs of the recognizers need to be augmented with a confidence measure.

The second strategy uses the performance of each individual classifier on the validation set as a confidence measure. This means that for all words in the output sequence of a classifier the same confidence score is taken.

The third strategy uses more elaborated confidence values, i.e., it takes the confidence values output by the recognizers into account. This is not an easy task, because the confidences of the individual recognition systems are not directly comparable. While the CTC and Vision Objects© systems output a normalized recognition score, the HMM-based system gives a likelihood, which is barely useful for combination. However, the most difficult problem is the Microsoft© recognizer, which only gives three discrete values on the text line level. We have solved the problem as follows. The range of confidence scores of each classifier is divided into several intervals. For each interval, the accuracy on the validation set is calculated. This accuracy is then taken as the final confidence score during testing. An example is shown in Fig. 5. In this example the interval $[-0.2, 0]$ of the log confidences of a CTC system has been divided into 40 sub-intervals. As can be seen the accuracy increases if the confidence increases. Surprisingly, for the lowest confidence interval there is a higher accuracy on the validation set. This is caused by words with a low frequency. Often such words are correct, but they have only small language model probabilities. Note that the curve is smoothed to avoid discontinuities at the borders of each interval.

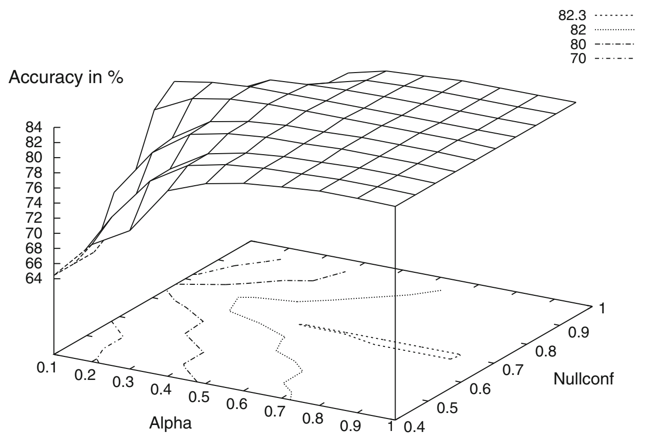


Fig. 6 Optimization of the parameters α and $C(\epsilon)$ (denoted as Nullconf)

The parameters for voting, α and $C(\epsilon)$, are optimized on the validation set for an ensemble of classifiers. The final voting strategy consists in taking the maximum rather than the average of the individual confidence scores in (1), because this strategy led to better recognition results in all experiments on the validation set. Figure 6 illustrates the accuracy of an example classifier ensemble on the validation set for different parameter combinations. It can be observed that the values do not significantly change for $\alpha > 0.8$. The highest accuracy in this example is reached at ($\alpha = 0.5$, $C(\epsilon) = 0.7$).

4.2.2 *Order of the classifiers*

The order of the classifiers has an influence on the alignment, for only a suboptimal alignment strategy is chosen. A commonly used strategy begins with the best classifier of the validation set and sequentially adds the next best classifier. This strategy has been adopted in this paper. This leads to the following sequence of recognition systems:

1. the on-line CTC systems
2. the Vision Objects© recognizer
3. the off-line CTC systems
4. the Microsoft© recognizer
5. the HMM-based recognizers.

4.2.3 *Classifiers included in the ensemble*

As explained previously (see also Table 1), there are 35 individual classifiers available for potential inclusion in the ensemble. However, it is well known that ensemble performance does not necessarily monotonically grow with ensemble size. Therefore, the question arises which of the individual classifiers actually to include in the ensemble. It can be expected that taking just the best recognition systems does not necessarily yield the optimal performance, because the CTC systems would be too dominant.

The following automatic strategy has been chosen to solve the problem. All possible combinations of up to six instances of each system have been validated automatically. Note that this is an implicit weighting for each of the non-CTC classifiers, because the same classifier can be included several times. The upper bound of six is motivated by experiments on the validation set where there was no performance increase when more than six classifiers have been used. The best combination on the validation set is as follows: five "extended on-line" CTC systems, one " on-line" CTC system, five Vision Objects© recognition systems, four off-line CTC systems, three Microsoft© recognition systems, and two "extended on-line" HMM-based system. The accuracy of this ensemble on the validation set is 83.34%, which is remarkably higher than the accuracy of the best individual classifier on the validation set, i.e., a CTC with only 76.60%.

Table 2 Results of the best combination on the test set

System	Accuracy (%)
Best individual	81.26
classifier	81.26
"voting 1"	85.88
"voting 2"	85.86
"voting 3"	86.16
Oracle	92.03

4.3 Test set results

After finding the optimal combination of recognizers on the validation set, the corresponding MCS is tested on the independent test set. The results on the test set appear in Table 2. The three different voting strategies described in Sect. 4.2.1 are denoted by "voting 1" for voting based on the number of occurrences, "voting 2" for voting using the accuracy on the validation set as a confidence measure, and "voting 3" for advanced voting taking also the confidence of the classifiers into account. The combination based on the number of occurrences performs statistically significantly better than the best individual classifier at a significance level of 0.1% (using a standard z-test). The recognition accuracy is 85.88%, resulting in a remarkable relative error reduction of 24.65%.

The second voting strategy, which uses the accuracy on the validation set as a confidence score, performs with 85.86%, which is almost as good as "voting 1". The third strategy leads to a relative error reduction of more than 26.14%, compared to the best individual classifier, which is significant at the 0.1% level. Note that the additional effort made by the more elaborated combination method also brings a statistically significant performance increase, compared to "voting 1".

The last row in Table 2 presents the result of an oracle voting. The oracle is assumed to know the transcription and always takes the correct word if at least one individual classifier recognized it correctly. Hence the oracle performance is an upper bound for the combination of the classifiers given the alignment. This result shows that with an improved voting strategy, a further improvement of about 6% is theoretically possible. On the other hand, one can argue that the accuracy of 86.16% achieved with "voting 3" is already quite close to the best possible value.

5 Conclusions

In this paper, a multiple classifier system (MCS) for the recognition of handwritten notes written on a whiteboard has been presented. The ROVER framework has been used to combine the output sequences of the individual recognizers. This framework first aligns the word sequences incrementally in a word transition network and then applies a voting strategy to select the final result at each output position.

HMMs and recursive neural networks are used as the base recognizers of the multiple classifier system. These base recognizers utilize different sets of on-line and off-line features extracted from the handwritten input. A bigram language model is included in each of these recognizers, which has been trained on the second validation set of the underlying database. Furthermore, commercial recognition systems from Microsoft© and Vision Objects© have been included in the combination. The first validation set of the database has been used to optimize the individual systems and the combination parameters.

Experiments have been performed on the IAM-OnDB-t2 benchmark. The recognition results achieved by the individual recognizers show that the CTC systems based on online features have the best accuracy, followed by the Vision Objects© recognizer. A possible explanation for this ranking is that the CTC systems has been trained on whiteboard data. The HMM-based systems have the lowest accuracy. Nonetheless, including one HMM-based recognizer in the ensemble leads to a performance increase.

The experimental results on the test set show a highly significant improvement of the recognition performance over the best individual classifier. The combination based on the number of occurrences performs with 85.88%, while the best individual classifier only achieves 81.26%. Two other voting strategies, which calculate a confidence score for each word class, have also been investigated in this paper. While using the recognition accuracy on the validation as a confidence score does not lead to an improved performance, a significantly higher performance has been achieved with an advanced approach. This approach takes also the confidence scores of the individual recognizers into account. The optimized combination performs with 86.16%, which represents a relative error reduction of 26.15% over the best individual classifier.

While the main goal of this research was to achieve higher recognition performance, an interesting aspect of the recognition system is the computation time needed for obtaining the recognition result. The individual recognizers differ in computation time, i.e., the commercial systems need about 500 ms, the neural networks need about 2 s and the HMMs need more than a minute for the recognition of one text line. Combining the systems leads to a longer computation time, for all individual times are accumulated. However, if a parallel computing architecture is available, the recognizers can be run in parallel. This results in no significant overhead as only a negligible time is needed for combination.

An interesting idea to reduce the operation time would be to run the classifiers in a specific order and only run the later classifiers if their results are needed in the combination scheme.[5] If the earlier classifiers can reach a decision with high confidence, then the results from the later classifiers may not be needed. This could lead to time savings without changing any of the recognition results.

In addition to the MCS recognition accuracy, the performance of an oracle system has been investigated. It performs with 92.02%, i.e., 6% higher than the performance of the best combination. This shows that there is still a high potential in the combination, which is a promising research topic for future work. For example, other confidence measures proposed in the literature [3,12,32] could be considered for advanced combination strategies.

Acknowledgments This work is supported by the Swiss National Science Foundation program "Interactive Multimodal Information Management (IM2)" in the Individual Project "Visual/Video Processing". The authors thank Roman Bertolami for providing tools for the combination. Furthermore, thanks are due to the IDSIA in Switzerland for providing the neural network toolkit.

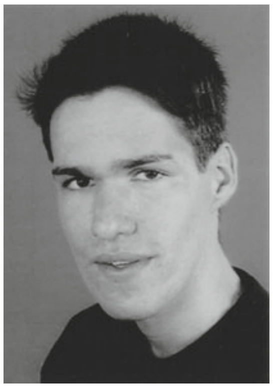
References

1. Bandoh, H., Nemoto, H., Sawada, S.I., Indurkhya, B., Nakagawa, M.: Development of educational software for whiteboard environment in a classroom. In: Int. Workshop on Advanced Learning Technologies, pp. 41–44 (2000)
2. Bauer, L.: Manual of Information to Accompany the Wellington Corpus of Written New Zealand English. Department of Linguistics, Victoria University, Wellington (1993)
3. Bertolami, R., Zimmermann, M., Bunke, H.: Rejection strategies for offline handwritten text line recognition. Pattern Recognit. Lett. **27**(16), 2005–2012 (2006)

[5] We would like to thank one of the anonymous reviewers for this remark.

4. Bunke, H.: Recognition of cursive Roman handwriting—past present and future. In: Proc. 7th. Int. Conf on Document Analysis and Recognition, vol. 1, pp. 448–459 (2003)
5. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. B **39**(1), 1–38 (1977)
6. Fernández, S., Graves, A., Schmidhuber, J.: Sequence labelling in structured domains with hierarchical recurrent neural networks. In: Proc. 20th Int. Joint Conf. on Artificial Intelligence, pp. 774–779 (2007)
7. Fink, G.A., Wienecke, M., Sagerer, G.: Video-based on-line handwriting recognition. In: Proc. 6th Int. Conf. on Document Analysis and Recognition, pp. 226–230 (2001)
8. Fiscus, J.: A post-processing system to yield reduced word error rates: recognizer output voting error reduction ROVER. In: IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 347–352 (1997)
9. Forney, G.D.: The Viterbi algorithm. Proc. IEEE **61**, 268–278 (1973)
10. Francis, W.N., Kucera H.: 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)
11. Friedland, G., Knipping, L., Schulte, J., Tapia, E.: E-Chalk: a lecture recording system using the chalkboard metaphor. Intell. J. Interact. Technol. Smart Education (ITSE) **1**, 9–20 (2004)
12. Gorski, N.: Optimizing error-reject trade off in recognition systems. In: Proc. 4th Int. Conf. on Document Analysis and Recognition, vol. 2, pp. 1092–1096 (1997)
13. Günter, S., Bunke, H.: HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. Pattern Recognit. **37**, 2069–2079 (2004)
14. Haeb-Umbach, R., Ney, H.: Improvements in beam search for 10,000-word continuous-speechrecognition. Trans. Speech Audio Process. **2**, 353–356 (1994)
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
16. Huang, Y.S., Suen, C.Y.: A method of combining multiple experts for the recognition of unconstrained handwritten numerals. IEEE Trans. Pattern Anal. Mach. Intell. **17**(1), 90–94 (1995)
17. Iwayama, N., Akiyama, K., Tanaka, H., Tamura, H., Ishigaki, K.: Handwriting-based learning materials on a tablet pc: a prototype and its practical studies in an elementary school. In: Proc. 9th Int. Workshop on Frontiers in Handwriting Recognition, pp. 533–538 (2004)
18. Jäger, S., Manke, S., Reichert, J., Waibel, A.: Online handwriting recognition: the NPen++ recognizer. Intell. J. Document Anal. Recognit. **3**(3), 169–180 (2001)
19. Johansson, S.: The tagged LOB Corpus: User's Manual. Norwegian Computing Centre for the Humanities, Norway (1986)
20. Knerr, S., Anisimov, V., Baret, O., Gorsky, N., Price, D., Simon, J.C.: The A2iA INTERCHEQUE system: Courtesy amount and legal amount recognition for French checks. Intell. J. Pattern Recognit. Artif. Int. **11**(4), 505–548 (1997)
21. Knerr, S., Augustin, E.: A neural network-hidden Markov model hybrid for cursive word recognition. Intell. Conf. Pattern Recognit. **2**, 1518–1520 (1998)
22. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley, New York (2004)
23. Lallican, P.M., Viard-Gaudin, C., Knerr, S.: From off-line to online handwriting recognition. In: Int. Workshop on Frontiers in Handwriting Recognition, pp. 303–312 (2000)
24. Liwicki, M., Bunke, H.: 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 (2005)
25. Liwicki, M., Bunke, H.: HMM-based on-line recognition of handwritten whiteboard notes. In: Proc. 10th Int. Workshop on Frontiers in Handwriting Recognition, pp. 595–599 (2006)
26. Liwicki, M., Bunke, H.: Combining on-line and off-line systems for handwriting recognition. Proc. 9th Intell. Conf. Document Anal. Recognit. **1**, 372–376 (2007)
27. Liwicki, M., Bunke, H.: Handwriting recognition of whiteboard notes—studying the influence of training set size and type. Intell. J. Pattern Recognit. Artif. Intell. **21**(1), 83–98 (2007)
28. Liwicki, M., Graves, A., Bunke, H., Schmidhuber, J.: A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. Proc. 9th Intell. Conf. Document Anal. Recognit. **1**, 367–371 (2007)
29. Moore, D.: The IDIAP smart meeting room. Technical report, IDIAP-Com (2002)
30. Munich, M.E., Perona, P.: Visual input for pen-based computers. In: Proc. 3rd Int. Conf. on Pattern Recognition, pp. 33–37 (1996)
31. Perraud, F., Viard-Gaudin, C., Morin, E.: Language independent statistical models for on-line handwriting recognition. In: Int. Workshop on Frontiers in Handwriting Recognition, pp. 435–440, (2006)
32. Pitrelli, J.F., Perrone, M.P.: Confidence-scoring post-processing for off-line handwritten-character recognition verification. Proc. 7th Intell. Conf. Document Anal. Recognit. **1**, 278–282 (2003)
33. Pittman, J.A.: Handwriting recognition: Tablet pc text input. Computer **40**(9), 49–54 (2007)
34. Plamondon, R., Srihari, S.N.: On-line and off-line handwriting recognition: a comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. **22**(1), 63–84 (2000)
35. Rojas, R.: Neural Networks—a Systematic Introduction. Springer, Heidelberg (1996)
36. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**, 2673–2681 (1997)
37. Tappert, C.C., Suen, C.Y., Wakahara, T.: The state of the art in online handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. **12**(8), 787–808 (1990)
38. Velek, O., Jäger, S., Nakagawa, M.: Accumulated-recognition-rate normalization for combining multiple on/off-line Japanese character classifiers tested on a large database. In: Proc. 4th Workshop on Multiple Classifier Systems, pp. 196–205 (2003)
39. Vinciarelli, A.: A survey on off-line cursive script recognition. Pattern Recognit. **35**(7), 1433–1446 (2002)
40. Vinciarelli, A., Perrone, M.: Combining online and offline handwriting recognition. In: Proc. 7th Int. Conf. on Document Analysis and Recognition, pp. 844–848 (2003)
41. Wagner, R., Fischer, M.: The string-to-string correction problem. J. ACM **21**, 168–173 (1974)
42. Waibel, A., Schultz, T., Bett, M., Malkin, R., Rogina, I., Stiefelhagen, R., Yang, J.: SMaRT: the smart meeting room task at ISL. Proc. IEEE ICASSP **4**, 752–755 (2003)
43. Wang, L., Jiang, T.: On the complexity of multiple sequence alignment. J. Comput. Biol. **1**(4), 337–348 (1994)
44. Xiangyun, Ye., Cheriet, M., Suen, C.Y.: StrCombo: combination of string recognizers. Pattern Recognit. Lett. **23**, 381–394 (2002)

Author biographies



Marcus Liwicki received his M.S. degree in Computer Science from the Free University of Berlin, Germany, in 2004, and his PhD degree from the University of Bern, Switzerland, in 2007. Currently he is a senior researcher and lecturer at the German Research Center for Artificial Intelligence (DFKI). His research interests include knowledge management, semantic desktop, electronic pen-input devices, on-line and off-line handwriting recognition and document analysis.

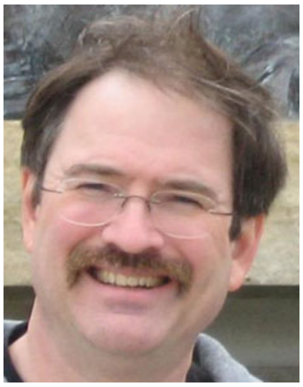
Marcus Liwicki is a member of the IAPR and frequent reviewer for international journals, including IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Audio, Speech and Language Processing, Pattern Recognition, and Pattern Recognition Letters. He is a member of the program committee of the IEEE ISM Workshop on Multimedia Technologies for E-Learning, and a reviewer of several IAPR conferences and the IGS conference. Marcus Liwicki is a co-author of the book "Recognition of Whiteboard Notes – Online, Offline, and Combination", published by World Scientific in October 2008. He has more than 20 publications, including seven journal papers.

Horst Bunke received his M.Sc. 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 to 1996, 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 served as first 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., Advisory Editor of Pattern Recognition, Associate Editor of Acta Cybernetica and Frontiers of Computer Science in China, and former Associate Editor of the International Journal of Document Analysis and Recognition, and Pattern Analysis and Applications.

Horst Bunke received an honorary doctor degree from the University of Szeged, Hungary, and 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, 1998-2007), the University of Nevada at Las Vegas (1994), Kagawa University, Takamatsu, Japan (1995), Curtin University, Perth, Australia (1999), and Australian National University, Canberra (2005).

He served as a co-chair of the fourth International Conference on Document Analysis and Recognition held in Ulm, Germany, 1997 and as a Track Co-Chair of the 16th and 17th International 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, a co-chair of the seventh IAPR Workshop on Document Analysis Systems held in Nelson, NZ, 2006, a co-chair of the tenth International Workshop on Frontiers in Handwriting Recognition, held in La Baule, France, 2006, and a program co-chair of the 11th International Conference on Frontiers in Handwriting Recognition held in Montreal, Canada, 2008. 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). Horst Bunke has more than 600 publications, including 40 authored, co-authored, edited or co-edited books and special editions of journals.



James A. Pittman received his M.S. and Ph.D. degrees in Industrial Engineering and Operations Research from Virginia Tech. In 1984 he joined MCC, a research consortium in Austin, Texas. He spent 9 years there working on a variety of recognition and human-computer-interface research topics, with more than half of that time spent in handwriting recognition using neural networks. He also experimented with neural networks for detecting credit card fraud.

In the early 1990s James spent 2 years at the National University of Singapore working on recognizing cursive Chinese script. He then spent 2 years at the Oregon Graduate Institute working on the integration of handwriting recognition with speech recognition, natural language processing, 3D graphics, and map presentation in a military training application.

James joined Microsoft in 1997, and has spent the past 12 years working on cursive handwriting recognition for the Tablet PC. In the late 1990s he focused on combining multiple classifiers. More recently he drove the expansion to Dutch, Portuguese, Swedish, Danish, Norwegian (both Bokmal and Nynorsk), Polish, Mexican Spanish, and Swiss German. His current research focuses on reducing the cost of expanding recognition to additional languages.

Stefan Knerr holds a M.S. in Theoretical Physics from the University of Heidelberg, Germany, and a Ph.D. from the University Pierre et Marie Curie in Paris, France, where he worked on advanced pattern recognition problems. He has more than 20 years of experience with handwriting recognition technology, and has been working for leading IT companies and international research institutes such as Siemens (Germany), Centre National de la Recherche Scientifique (France) and the International Computer Science Institute in Berkeley (USA). He has also played a key role in building the software company A2iA (France), market leader for bank check recognition, and leading it to success.

In 1998 he founded Vision Objects in Nantes, France, to develop the next generation of advanced multi-lingual natural handwriting recognition software using innovative algorithms from the fields of pattern recognition, computational linguistics, neural computation, and artificial intelligence. Vision Objects has become a worldwide provider and industry leader in handwriting recognition technology for pen-based user interfaces and data entry automation.