The Writer Independent Online Handwriting Recognition System *frog on hand* and Cluster Generative Statistical Dynamic Time Warping

Claus Bahlmann and Hans Burkhardt, *Member*, *IEEE*

Abstract—In this paper, we give a comprehensive description of our writer-independent online handwriting recognition system *frog on hand*. The focus of this work concerns the presentation of the classification/training approach, which we call *cluster generative statistical dynamic time warping (CSDTW)*. CSDTW is a general, scalable, HMM-based method for variable-sized, sequential data that holistically combines cluster analysis and statistical sequence modeling. It can handle general classification problems that rely on this sequential type of data, e.g., speech recognition, genome processing, robotics, etc. Contrary to previous attempts, clustering and statistical sequence modeling are embedded in a single feature space and use a closely related distance measure. We show character recognition experiments of *frog on hand* using CSDTW on the UNIPEN online handwriting database. The recognition accuracy is significantly higher than reported results of other handwriting recognition systems. Finally, we describe the real-time implementation of *frog on hand* on a Linux Compaq iPAQ embedded device.

Index Terms—Pattern recognition, handwriting analysis, Markov processes, dynamic programming, clustering.

1 Introduction

During recent years, the task of online handwriting recognition (HWR) has gained an immense importance in every day applications, mainly due to the increasing popularity of the personal digital assistant (PDA). Currently, a new generation of *smart phones* and *tablet PCs*, which also rely on handwriting input, is targeting the consumer market. However, in the majority of these devices, the handwriting input method is still not satisfactory. In current PDAs, people use input methods which differ from the natural writing habit, e.g., the widespread *Graffiti*.[1] Other systems use a more natural input; however, they still rely on restricted writing styles.

The difficulty of designing a writer independent system is commonly explained as follows: First, small devices like PDAs and smart phones have limitations in computational power and memory size which is cumbersome in the system design for these devices. Second, for a writer independent solution, the system has to discriminate between a large variety of different writing styles which are present in the target group of users. Even more difficult for online recognition, a writing which looks similar in a graphical (i.e., offline) representation, can have a different sequential (i.e., online) representation.

Thus, there is demand for a handwriting recognition system which is efficient, scalable to the device's capability, accurate, and which can deal with the natural handwriting of a wide range of different writers and writing styles.

In this contribution, we give a detailed description of *frog on hand* (*f*reiburg *r*ec*og*nition of *on*line *hand*writing), an online handwriting recognition system we have developed during the last few years. The main novel aspects compared to other systems [6], [11], [13], [15], [29], [30], [31], [33], [39] include the following.

We have developed the novel learning approach *cluster generative statistical dynamic time warping* (CSDTW), which—based on dynamic time warping (DTW) and hidden Markov modeling (HMM)—treats writing variations by *holistically* combining cluster analysis and generative statistical sequence modeling.

The aid of cluster analysis for modeling sequential, especially handwriting data, has been also addressed by other researchers. Some used hierarchical clustering methods [19], [37], [38], others k-means clustering [24] or hybrid solutions [27]. The approach proposed in this paper is different from recent approaches with respect to the following issue. Most of these follow mainly two different philosophies when integrating the clustering into the classifier:

1. A powerful classifier (e.g., an HMM) uses cluster information which has been revealed in a different feature space and with a different assumption about (dis-) similarity [7], [19], [21], [35]. Hence, those algorithms cannot be certain that the clusters found in the cluster space correspond to well-formed clusters in the classifier space.
2. Clustering and classification are performed in the same feature space, however, the classifier uses simple, limited techniques like template matching [27], [37], [38].

One deviation from these two philosophies is known to the authors. Perrone and Connell [24] embed a clustering/HMM hybrid in the single feature space of the HMM parameters.

[1] http://www.palm.com/products/input/.

• *The authors are with the Computer Science Department, Albert-Ludwigs-University Freiburg, 79110 Freiburg, Germany.*
E-mail: {bahlmann, burkhardt}@informatik.uni-freiburg.de.

Manuscript received 22 Apr. 2003; revised 17 Sept. 2003; accepted 20 Sept. 2003.
Recommended for acceptance by V. Govindaraju.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 0048-0403.

0162-8828/04/$20.00 © 2004 IEEE Published by the IEEE Computer Society

However, this approach lacks a generic quality, since their (iterative) clustering relies on a reasonable initialization, which the authors perform by a manual adjustment.

In the proposed CSDTW solution, both clustering and statistical modeling are embedded in a single feature space—the space of the training samples. As will be shown, they assume a closely related distance measure, while retaining a powerful, HMM-based classification scheme. Additionally, the CSDTW approach includes a strategy to an open question in statistical sequence modeling research—the problem of topology selection and initializing the statistical models in the context of the commonly used iterative classifier training. Thus, no manual topology selection or initialization has to be employed.

Another attractive property of CSDTW is its scalability. By adjusting particular parameters, the system designer can straightforwardly find a compromise between the classifier size and the recognition accuracy.

We shall start with a description of the underlying handwriting data, our preprocessing, and feature selection in the section that follows. Section 3 covers the classification and training of CSDTW. In Section 4, experimental results of CSDTW on the UNIPEN [12] online handwriting database and a comparison to UNIPEN results of other recognition systems are presented. In Section 5, we give a brief description of the implementation of *frog on hand* on a Linux Compaq iPAQ PDA. Section 6 concludes this contribution.

2 The Online Handwriting Recognition Context

Handwriting recognition (HWR) is the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation [26]. Online HWR refers to the situation where the recognition is performed concurrently to the writing process.

2.1 Online Handwriting Data

Online handwriting data is typically a dynamic, digitized representation of the pen movement, generally describing sequential information about position, velocity, acceleration, or even pen angles as a function of time. In recent years, the UNIPEN database [12] has become the most popular publicly available data collection in online handwriting research.

UNIPEN represents a writing curve as a series of so-called *pen-down* and *pen-up* components. Each component contains a sequence of pen tip information which is sampled from the writer's pen movement, usually (but not exclusively) with regular interval in time. With *frog on hand*, we solely refer to the horizontal and vertical coordinates $\mathbf{p}_i = (x_i, y_i)^T$.[2] Pen-down components are recorded when the pen tip touches the surface, pen-up components when it is lifted (see Fig. 1a). In this paper, we focus on the situation where a writing represents an isolated character. General methods exist that extend the character recognition onto a word-level basis [8], [20], [26]. We are pursuing this issue in our current work [34].

2.2 Data Preprocessing

Preprocessing usually addresses the problems of data reduction, elimination of imperfections, and normalization [10]. We utilize two simple steps, both eliminating irrelevant and disturbing coordinates.

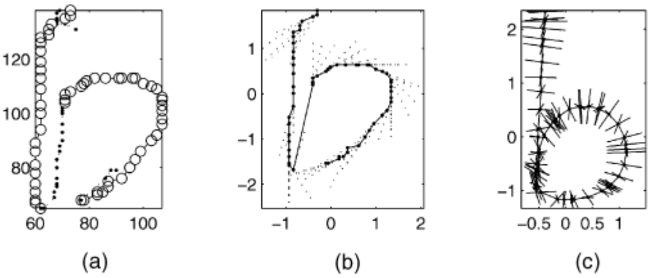


Fig. 1. (a) A sample of a UNIPEN character P of class "b". Coordinates are illustrated by a circle for pen-down and by a point for pen-up components. (b) Illustration of a feature sequence t and the features $\tilde{x}_i$, $\tilde{y}_i$, and θ_i: $\tilde{x}_i$ and $\tilde{y}_i$ are plotted according to their value in the $\tilde{x}$-$\tilde{y}$-plane. The dotted lines illustrate θ_i by the direction of the tangent. (c) An example of an allograph reference model $\mathcal{R}$ (of character class "b"). The reader can see a connected sequence of dots, each one with two additional lines attached. As a reference model is represented by a sequence of Gaussian probability density functions (PDFs), each dot illustrates the mean of the Gaussian and the lines the covariance matrix. The direction of the two lines match the projection of the first two eigenvectors onto the $\tilde{x}$-$\tilde{y}$-plane, their length the square root of the corresponding eigenvalues. Thus, the lines indicate the orientation and width of the Gaussian.

Removal of pen-up components. The writing's pen-up components are eliminated. The remaining pen-down components are concatenated into a single coordinate sequence $P = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$.

Removal of repeated samples. The UNIPEN database includes a number of writings that contain repetitions of coordinates, i.e., $\mathbf{p}_i = \mathbf{p}_{i+1} = \ldots = \mathbf{p}_{i+d}$ for some i and d. When computing differential features, these co-occurrences can cause disturbing singularities. Thus, we remove the extra occurrences $\mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+d}$.

2.3 Feature Selection

In our feature selection, a (preprocessed) character $P = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$ is transformed into a sequence $\boldsymbol{t} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]$ of feature vectors $\mathbf{t}_i = (t_{i1}, \ldots, t_{iF})^T \in \mathbb{R}^F$.

We have studied several local features, inspired by recent publications [11], [13], [15], [30], [33]. Those studies include a normalized representation of the coordinates, a representation of the tangent slope angle, a normalized curvature, the ratio of tangents, etc. We observed the best character recognition rates—in combination with our classification—using the following selection:

t_{i1}, t_{i2}: **normalized horizontal and vertical coordinates**. $t_{i1} = \tilde{x}_i = \frac{x_i - \mu_x}{\sigma_y}$ and $t_{i2} = \tilde{y}_i = \frac{y_i - \mu_y}{\sigma_y}$ are the pen coordinates normalized by the sample mean $\boldsymbol{\mu} = (\mu_x, \mu_y)^T = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i$ and (vertical) y standard deviation

$$\sigma_y = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\mu_y - y_i)^2}$$

of the character's sample points.

t_{i3}: **tangent slope angle**. $t_{i3} = \theta_i = \arg((x_{i+1} - x_{i-1}) + J \cdot (y_{i+1} - y_{i-1}))$, with $J^2 = -1$ and "arg" the phase of the complex number above, is an approximation of the tangent slope angle at point i.

[2] Additional information like velocity, acceleration, and pen angles is included in the UNIPEN specification, but not present in each database sample.

Since θ_i is a directional quantity, a special treatment for the computation of probabilities, statistics, and distances is necessary [2].

To summarize, a feature vector sequence is defined as $\boldsymbol{t} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]$, each vector of it as $\mathbf{t}_i = (\tilde{x}_i, \tilde{y}_i, \theta_i)^T$. Fig. 1b gives a graphical illustration of the feature representation.

3 CSDTW

For a robust classification, we deploy a holistic combination of a DTW-based cluster analysis and an HMM-based statistical modeling of generated clusters. We call the approach *cluster generative statistical dynamic time warping* (CSDTW). In this combination, both elements share a closely related distance measure, thus the term "holistic." Indeed, this aspect discriminates CSDTW from previous attempts of combining clustering and statistical modeling [7], [19], [21], [35].

In the context of handwriting recognition, the philosophy of CSDTW is to model each different character's writing style by a distinct statistical model. In the literature, the terms *allograph* [6], [24], [25], [27], [31], [38] or *lexeme* [7] have been evolved to describe such a distinct character style. With CSDTW, the scope of the term allograph can be flexibly implemented. Adaptable to the classification background, CSDTW includes a mechanism to specify whether just large data variations (e.g., culturally conditioned) or additionally smaller variations (e.g., due to different writer habits or the word context in that a character is written) shall be regarded for the generation of the distinct allograph models.

3.1 CSDTW Classification

The aim of this section is to describe the classification part of CSDTW. At this point, we assume that a classifier has already been trained from a data set of labeled characters. In order to fix the notation, we first give a brief review of the DTW distance before we introduce the statistical DTW (SDTW) distance and, finally, the cluster generative SDTW (CSDTW) classification scheme.

3.1.1 Dynamic Time Warping (DTW) Distance

DTW is a concept that allows an elastic match of two sequences. Details are described in literature, e.g., in the textbook of Rabiner and Juang [28, chapter 4.7]. Here, we want to review the basic concepts.

Suppose that the following is given:

- two vector sequences $\boldsymbol{t} = [\mathbf{t}_1, \ldots, \mathbf{t}_{N_t}]$ and $\boldsymbol{r} = [\mathbf{r}_1, \ldots, \mathbf{r}_{N_r}]$ with $\mathbf{t}_i, \mathbf{r}_j \in \mathbb{R}^F$; one can think of $\boldsymbol{t}$ being a "test" and $\boldsymbol{r}$ being a "reference" sequence,
- a so-called *alignment* (or *warping*) *path* $\boldsymbol{\Phi} = (\phi(1), \ldots, \phi(N))$ with

  $$\phi = (\phi_t, \phi_r) : \{1, \ldots, N\} \to \{1, \ldots, N_t\} \times \{1, \ldots, N_r\},$$

  the purpose of which is to define an alignment of corresponding regions in $\boldsymbol{t}$ and $\boldsymbol{r}$, and
- a *local* distance function $d : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}$ of two sequence elements

Then, the *alignment distance* $D_{\boldsymbol{\Phi}}[d](\boldsymbol{t}, \boldsymbol{r})$ is defined as the mean local distance of $\boldsymbol{t}$ and $\boldsymbol{r}$ with respect to d and the particular alignment path $\boldsymbol{\Phi}$

$$D_{\boldsymbol{\Phi}}[d](\boldsymbol{t}, \boldsymbol{r}) = \frac{1}{N} \sum_{n=1}^{N} d\big(\mathbf{t}_{\phi_t(n)}, \mathbf{r}_{\phi_r(n)}\big). \tag{1}$$

Regard that the parameterization of $D_{\boldsymbol{\Phi}}$ by d is quite unusual in literature, however, it will prove to be practical in the remainder of this contribution (particularly in Sections 3.1.2, 3.1.3, and 3.1.4).

Further, the *DTW (Viterbi) distance* $D^*[d](\boldsymbol{t}, \boldsymbol{r})$ is defined as the alignment distance according to the *Viterbi path* $\boldsymbol{\Phi}^*$. The Viterbi path is the *optimal* alignment path in the sense that it minimizes $D_{\boldsymbol{\Phi}}[d](\boldsymbol{t}, \boldsymbol{r})$:

$$D^*[d](\boldsymbol{t}, \boldsymbol{r}) = D_{\boldsymbol{\Phi}^*}[d](\boldsymbol{t}, \boldsymbol{r}) = \min_{\boldsymbol{\Phi}} \{D_{\boldsymbol{\Phi}}[d](\boldsymbol{t}, \boldsymbol{r})\}. \tag{2}$$

In situations when we want to emphasize that a specific Viterbi path comes from the alignment of a particular $\boldsymbol{t}$ and $\boldsymbol{r}$, we will write $\boldsymbol{\Phi}^*_{\boldsymbol{t},\boldsymbol{r}}$ instead of $\boldsymbol{\Phi}^*$.

It is convenient to model $\boldsymbol{\Phi}$ as a sequence of *transitions* from a transition set $\mathbb{P}$, i.e., $\Delta\phi = \phi(n+1) - \phi(n) \in \mathbb{P}$, $n = 1, \ldots, N-1$. We use the ones that are known as *Sakoe-Chiba transitions* in literature [28, chapter 4.7]. These only allow forward steps of size 1 in $\boldsymbol{t}$, $\boldsymbol{r}$, or in both of them, i.e.,

$$\mathbb{P} = \{(1,0), (0,1), (1,1)\}. \tag{3}$$

The alignment paths are constrained to include the endpoints of both $\boldsymbol{t}$ and $\boldsymbol{r}$, i.e., $\phi(1) = (1,1)$ and $\phi(N) = (N_t, N_r)$.

In literature, often the Euclidean distance

$$d\big(\mathbf{t}_i, \mathbf{r}_j\big) = \big\|\mathbf{t}_i - \mathbf{r}_j\big\| \tag{4}$$

or its square is used for the local distance of the sequence elements. However, we will substitute (4) by a distance that is specifically adapted to CSDTW's combination of clustering and statistical modeling, namely,

$$\tilde{d}\big(\mathbf{t}_i, \mathbf{r}_j\big) = \frac{1}{2}\Big(\ln(|2\pi\Sigma|) + \big(\mathbf{t}_i - \mathbf{r}_j\big)^T \Sigma^{-1} \big(\mathbf{t}_i - \mathbf{r}_j\big)\Big) + \ln(|\mathbb{P}|). \tag{5}$$

Here, $|\mathbb{P}|$ denotes the cardinality of the transition set $\mathbb{P}$, Σ a global covariance matrix of dimension $F \times F$, and $|\Sigma|$ its determinant. Σ can be used to model prior knowledge, e.g., about expected variances σ_f^2 of the feature dimension f, i.e., $\Sigma = \mathrm{diag}\big(\sigma_1^2, \ldots, \sigma_F^2\big)$. Alternatively, if no prior knowledge is given, it can be set to unity, i.e., $\Sigma = \boldsymbol{I}$, or a scalar multiple of it.

A motivation for this particular choice of $\tilde{d}$ will follow by the deliberations of Sections 3.1.2 and 3.1.4.

It can (not trivially) be shown [3] that the distance $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ violates the triangle inequality, thus is not a metric. However, it is symmetric.

3.1.2 Statistical DTW (SDTW) Distance

We introduce the statistical DTW (SDTW) distance as an extension of the DTW distance, which uses a very similar framework as that of (1), (2), and (3). However, it additionally embeds a statistical modeling of each feature subspace $\mathbb{R}^F$ and the transitions $\Delta\phi$ by a modification of the reference template representation and the local distance.

The SDTW formulation will be shown to be equivalent to HMM with a specific transition modeling. However, in our context, the described SDTW formulation has the benefit over the commonly used HMM formulation that the connection to the DTW distance can directly be realized. In this sense, the

DTW distance with the local distance of (5) will turn out to be an exact specialization of the SDTW distance.

In the SDTW modeling, a character reference will not be represented by a sequence $\boldsymbol{r} = [\mathbf{r}_1, \ldots, \mathbf{r}_{N_r}]$ of *feature vectors*, as with DTW, but by a sequence $\mathcal{R} = [\mathbf{R}_1, \ldots, \mathbf{R}_{N_\mathcal{R}}]$ of a tuple $\mathbf{R}_j$ of *statistical quantities*.

These comprise, equivalent to HMMs:

1. discrete probabilities, say $\alpha_j : \mathbb{P} \rightarrow [0, 1]$, for the statistical modeling of the transitions $\Delta\phi \in \mathbb{P}$ reaching the sequence's sample point j and
2. a continuous probability density function (PDF) $\beta_j : \mathbb{R}^F \rightarrow \mathbb{R}$, that is aimed to model the feature distribution $\mathbf{x} \in \mathbb{R}^F$ at the sequence's sample point j, i.e., $\beta_j(\mathbf{x}) = p(\mathbf{x}|\phi_\mathcal{R}(n) = j)$,

thus $\mathbf{R}_j = (\alpha_j, \beta_j)$.

We choose to model the PDF β_j by a unimodal, multivariate Gaussian, i.e.,

$$\begin{aligned}\beta_j(\mathbf{x}) &= \mathcal{N}_{\boldsymbol{\mu}_j, \Sigma_j}(\mathbf{x}) \\ &= \left(|2\pi\Sigma_j| \exp\left(\left(\mathbf{x} - \boldsymbol{\mu}_j\right)^T \Sigma_j^{-1}\left(\mathbf{x} - \boldsymbol{\mu}_j\right)\right)\right)^{-1/2}.\end{aligned} \tag{6}$$

A graphical illustration for an example of $\mathcal{R}$—a reference model of a character "b"—is shown in Fig. 1c.

In a general context, the assumption of such a type of PDF would be quite restrictive. Other HWR systems utilize Gaussian mixture or discrete probability models [13], which are more flexible than unimodal Gaussians. However, our approach assumes that large, especially, multimodal variations in the data are due to a different writing *style*. In this sense, the philosophy is to model each style by a distinct, but simple character subclass model.

With the new representation of the reference models compared to DTW—i.e., $\mathcal{R}$ instead of $\boldsymbol{r}$—an adapted local distance can be derived for the use in the SDTW context, starting from a Viterbi-path optimized maximum likelihood (ML) classification perspective; i.e., we want to choose that class that maximizes the likelihood $p(\boldsymbol{t}, \boldsymbol{\Phi}^*|\mathcal{R})$.

For the derivation of this new local distance it is assumed that the statistical quantities α_j and β_j are sequentially independent (which is contestable, but a common practice in sequence modeling [28], [32]) and each of the PDFs can be modeled by a Gaussian of (6).

Employing these assumptions, it can be shown that with the adapted local distance $\hat{d}$ given as

$$\begin{aligned}\hat{d}(\mathbf{t}_i, \mathbf{R}_j) = &\frac{1}{2}\left(\ln(|2\pi\Sigma_j|) + \left(\mathbf{t}_i - \boldsymbol{\mu}_j\right)^T \Sigma_j^{-1}\left(\mathbf{t}_i - \boldsymbol{\mu}_j\right)\right) \\ &- \ln(\alpha_j(\Delta\phi))\end{aligned} \tag{7}$$

the *SDTW (Viterbi) distance* $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$ that is defined in equivalence to (1) and (2) by

$$D_{\boldsymbol{\Phi}}[\hat{d}](\boldsymbol{t}, \mathcal{R}) = \frac{1}{N}\sum_{n=1}^{N} \hat{d}(\mathbf{t}_{\phi_t(n)}, \mathbf{R}_{\phi_\mathcal{R}(n)}) \tag{8}$$

$$D^*[\hat{d}](\boldsymbol{t}, \mathcal{R}) = D_{\boldsymbol{\Phi}^*}[\hat{d}](\boldsymbol{t}, \mathcal{R}) = \min_{\boldsymbol{\Phi}}\left\{D_{\boldsymbol{\Phi}}[\hat{d}](\boldsymbol{t}, \mathcal{R})\right\} \tag{9}$$

is related to $p(\boldsymbol{t}, \boldsymbol{\Phi}^*|\mathcal{R})$ by the equation

$$D^*[\hat{d}](\boldsymbol{t}, \mathcal{R}) = -\ln(p(\boldsymbol{t}, \boldsymbol{\Phi}^*|\mathcal{R})). \tag{10}$$

Thus, a minimum $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$ implies a maximum likelihood $p(\boldsymbol{t}, \boldsymbol{\Phi}^*|\mathcal{R})$.

A detailed derivation is given elsewhere [3]. A sketch of this derivation is as follows: We start with a decomposition of

$$\begin{aligned}p(\boldsymbol{t}, \boldsymbol{\Phi}^*|\mathcal{R}) &= p(\boldsymbol{t}|\boldsymbol{\Phi}^*, \mathcal{R})P(\boldsymbol{\Phi}^*|\mathcal{R}) \\ &= \prod_{n=1}^{N} \beta_{\phi^*_\mathcal{R}(n)}\left(\mathbf{t}_{\phi^*_t(n)}\right) \prod_{n=1}^{N} \alpha_{\phi^*_\mathcal{R}(n)}(\Delta\phi(n)),\end{aligned}$$

employ the Gaussian assumption of (6) and insert the result into (10). Finally, followed by a few straightforward transformations and the aid of (8) and (9), the validity of (7) can be realized.

3.1.3 Cluster Generative SDTW (CSDTW) Classification

Finally, this section explains the CSDTW classification. It shall be assumed that a superset

$$\Re = \left\{\mathcal{R}^{lk}\right\}_{l\in\{1,\ldots,L\}, k\in\{1,\ldots,K_l\}}$$

of SDTW subclass models $\mathcal{R}^{lk} = [\mathbf{R}^{lk}_1, \ldots, \mathbf{R}^{lk}_{N_{\mathcal{R}^{lk}}}]$ and a test pattern $\boldsymbol{t}$ (that corresponds to an isolated character in the HWR context) are given. Each $\mathcal{R}^{lk}$ is aimed to be a generative model for one "compact" cluster in the feature space of class l, hence, representing an allograph in the context of HWR. It will be explained in Section 3.2, how to get reasonable estimates for $\mathcal{R}^{lk}$.

In general, classification denotes the assignment of an estimated class label $\widehat{l}$ for the test pattern $\boldsymbol{t}$, the true class label of which is unknown. According to (10), a maximum likelihood (ML) classification principle is employed, when the classifier decision $\widehat{l}$ is given by

$$\widehat{l} = \operatorname*{argmin}_{l\in\{1,\ldots,L\}} \min_{k\in\{1,\ldots,K_l\}} \left\{D^*[\hat{d}](\boldsymbol{t}, \mathcal{R}^{lk})\right\}. \tag{11}$$

In this respect, (11) in combination with (3), (7), (8), and (9) define the framework for the CSDTW classification. If required, also classification confidence values in terms of a likelihood can be obtained with help of (10).

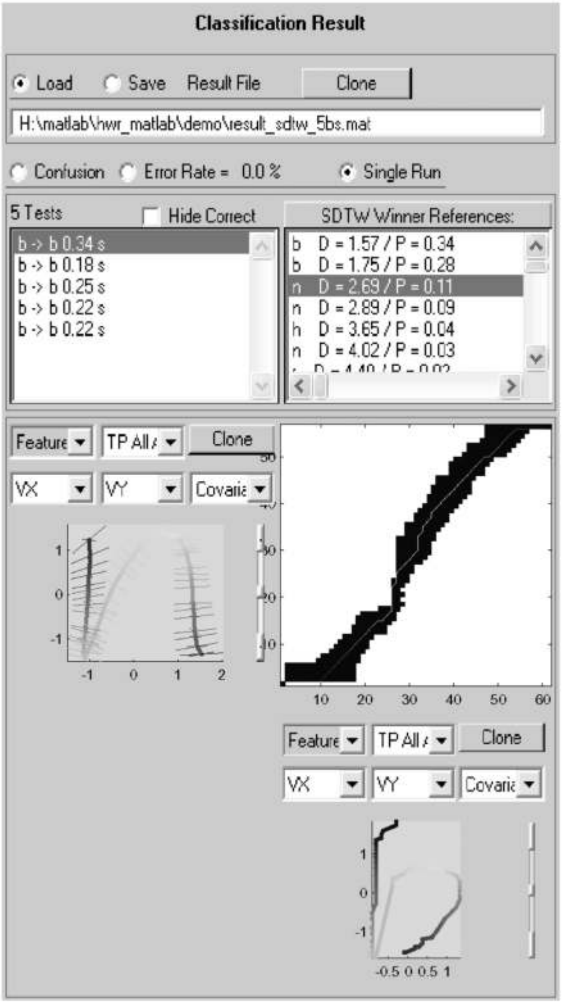
Fig. 2 shows a graphical illustration of a sample character classification.

3.1.4 DTW, SDTW, and HMM—A Unifying View

The concepts of DTW, SDTW, and HMM share common aspects, which will be pointed out in the following.

DTW and SDTW. A valuable insight concerning DTW and SDTW is given by the observation that the particular DTW Viterbi distance $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ is an exact specialization of the SDTW Viterbi distance $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$, when the simple identifications $\mathbf{r}_j = \boldsymbol{\mu}_j$, $\Sigma = \Sigma_j$, and $\alpha_j(\Delta\phi) = 1/|\mathbb{P}|$, $\forall\Delta\phi \in \mathbb{P}$ are used. With these, the matching of (5) and (7) can easily be verified. Further, the warping constraints defined by $\mathbb{P}$ do not differ between DTW and SDTW, resulting in an exact match of $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ and $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$ in the case of the identifications seen above. The only difference of $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ and $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$ is the statistical treatment of the feature space $\mathbb{R}^F$ and the transitions $\mathbb{P}$ for each sample point with SDTW. These deliberations justify the choice of the local distance of (5) in our DTW formulation.

SDTW and HMM. The formulation of SDTW is equivalent to HMMs. In the context of a standard HMM nomenclature, given e.g., by Rabiner and Juang [28], the HMM *states* q_j, *state*



Classification Result

H:\matlab\hwr_matlab\demo\result_sdtw_5bs.mat

Error Rate = 0.0 %

5 Tests

b -> b 0.34 s
b -> b 0.18 s
b -> b 0.25 s
b -> b 0.22 s
b -> b 0.22 s

SDTW Winner References:

b D = 1.57 / P = 0.34
b D = 1.75 / P = 0.28
n D = 2.69 / P = 0.11
n D = 2.89 / P = 0.09
h D = 3.65 / P = 0.04
n D = 4.02 / P = 0.03

Fig. 2. The classification of a test pattern t (of class “b,” illustrated in the lower right corner) . The list on the right shows labels of reference models $\mathcal{R}^{lk}$, sorted by the CSDTW Viterbi distance $D^*[\hat{d}](t, \mathcal{R}^{lk})$. Since a “b” is on top of the list, t is correctly classified. The rectangular area below the list illustrates the alignment of t with the third best match reference (a reference for the character “n,” illustrated to the left). White areas indicate regions, which were pruned by a beam search criterion (cf. Section 3.1.5). The Viterbi alignment path is illustrated by the sketched line: It traverses all aligned sample point pairs. Corresponding points in the “b,” “n,” and the Viterbi path are coded in the same (gray scale) color. The reader can see apparent temporal distortions at the beginning, the center, and the end region of the Viterbi alignment. The CSDTW Viterbi distance is $D^*[\hat{d}](t, \mathcal{R}^{lk}) = 2.69$ for this particular alignment.

transition probabilities a_{ij} and *observation functions* $b_j(\mathbf{o})$ have correspondences in the reference sequence index j, $\alpha_j(\Delta\phi)$ and $\beta_j(\mathbf{x})$, respectively. The starting point constraint $\phi(1) = (1, 1)$ (cf. Section 3.1.1) corresponds to the particular *HMM state prior probabilities* $\pi = (1, 0, 0, \ldots, 0)$.

However, to achieve full equivalence of SDTW and HMM, the concept of *null transitions* must be included into the HMM framework. Null transitions in HMMs allow a step in the state sequence without the emission of an observation. Hence, they correspond to the transition $\Delta\phi = (0, 1)$ in the SDTW distance computation. They have been introduced in HMMs of the IBM speech recognizer [1], but most common HMM implementations in handwriting recognition systems do not employ this concept. Instead, typically *linear* (corresponding to $\mathbb{P} = \{(1, 0), (1, 1)\}$) or *Bakis* (corresponding to $\mathbb{P} = \{(1, 0), (1, 1), (1, 2)\}$) transition models [28] are assumed in these systems.

For SDTW, the specific choice of the Sakoe-Chiba transitions $\mathbb{P} = \{(1, 0), (0, 1), (1, 1)\}$ and, thus, a “null-transition”-like modeling is very important for the following reasons:

1. Because of the flexibility of the alignment paths, the length of the reference sequences $\mathcal{R}$ in SDTW can be *arbitrary*. This is distinct from HMMs without null transitions, e.g., with $\mathbb{P} = \{(1, 0), (1, 1)\}$. There, the model length must not exceed the length of the test pattern due to the compulsory step in the test sequence. The arbitrariness of the reference length will allow an automatic, data-driven topology selection of the CSDTW reference models, as will be shown in Section 3.2.2.
2. Since DTW and SDTW are defined on the same set of transitions $\mathbb{P}$, the global warping capability does not differ between them. This behavior is indispensable in a scenario where DTW and SDTW shall model a related distance measure. In fact, CSDTW uses the DTW distance for clustering (cf. Section 3.2.1) and the SDTW distance during classification (cf. Section 3.1.3) and statistical parameter estimation (cf. Section 3.2.2).
3. Contrary to the linear or Bakis transitions, the chosen set $\mathbb{P} = \{(1, 0), (0, 1), (1, 1)\}$ is symmetric. Consequently, the Viterbi path $\mathbf{\Phi}^*$ is neither biased to a direction toward the test nor the reference pattern. This property is, in particular, convenient for the definition of a symmetric distance measure $D^*[\bar{d}](\mathcal{R}, \mathcal{R}')$ between two CSDTW reference models $\mathcal{R}$ and $\mathcal{R}'$, as we have recently studied [4].

In this section, we have illuminated the connections between SDTW and HMM. These considerations should help to transfer further development from one of these concepts to the other one. In particular, HMM refinements like state tying, state duration modeling, etc., [28] can straightforwardly be applied to SDTW and, thus, CSDTW.

3.1.5 Implementation Related Issues

When optimizing (2), (9), and (11), it is common and advantageous to apply techniques like dynamic programming and beam search in order to achieve a complexity reduction.

Dynamic programming. Dynamic programming (DP) is exhaustively described in literature [28], [32]. Its basic idea is to recursively develop the Viterbi matrix (i.e., the matrix $\mathbf{D}^* = [D^*_{ij}]$ of the prefixes' Viterbi distances $D^*_{ij} = D^*[d]([\mathbf{t}_1, \ldots, \mathbf{t}_i], [\mathbf{r}_1, \ldots, \mathbf{r}_j])$ (cf. Fig. 3a), exploiting Bellmann's *optimality principle*. The recursion can practically be implemented in correspondence with various geometrical schemes, e.g., by a row, column, or diagonal-wise expansion of $\mathbf{D}^*$. As the resulting computational cost is proportional to the Viterbi matrix size, DP reduces the exponential complexity of naively solving (2) to $\mathcal{O}(|\mathbb{P}|\tilde{N}^2)$, with $\tilde{N}$ the average sequence length [28, chapter 4.7.1].

It is worth noting that with DP only the unnormalized variant of the alignment distance, i.e., $D'_{\mathbf{\Phi}}[d](\boldsymbol{t}, \boldsymbol{r}) = \sum_{n=1}^{N} d(\mathbf{t}_{\phi_t(n)}, \mathbf{r}_{\phi_r(n)})$, can be minimized. This is due to the fact that the normalization factor N cannot be considered during DP, as it depends on the result of the DP search. Hence, our proceeding is to first compute $(D')^*[d](\boldsymbol{t}, \boldsymbol{r})$ with DP and

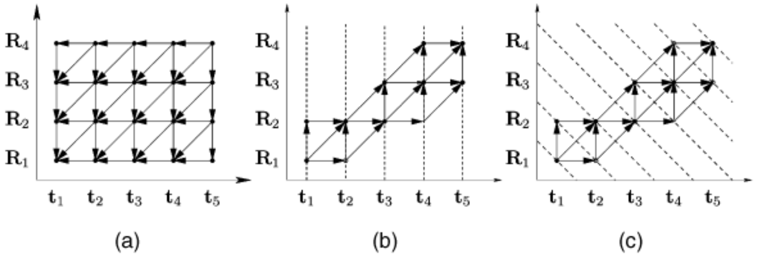


Fig. 3. (a) Dynamic programming. The Viterbi matrix $\mathbf{D}^*$ is recursively computed, indicated by the backward arrows. (b) A typical beam search scheme. At each step, only promising hypotheses are developed, indicated by the forward arrows. Here, the search is performed frame-synchronous, i.e., concurrent hypotheses are pruned along a time frame (the dashed vertical lines). (c) In CSDTW, the beam search is performed synchronous with respect to the matrix diagonals, in order to account for the symmetry of the transitions $\mathbb{P}$.

then normalize $D^*[d](\boldsymbol{t},\boldsymbol{r}) = \frac{1}{N}(D')^*[d](\boldsymbol{t},\boldsymbol{r})$ afterward. As a consequence, however, $D^*[d](\boldsymbol{t},\boldsymbol{r}) \neq \min_{\boldsymbol{\Phi}}\{D_{\boldsymbol{\Phi}}[d](\boldsymbol{t},\boldsymbol{r})\}$ in general with this DP solution (see also [28, chapters 4.7.2 and 4.7.3] for a discussion).

Beam search. Beam search [32], [16] denotes the strategy of eliminating path hypotheses, which are unlikely to turn out as the final, optimal solution at a specific level in the DP search. The hypotheses remaining from the elimination are called *active hypotheses*. This technique can further reduce the computational complexity of (2) to $\mathcal{O}(|\mathbb{P}|\tilde{N})$ (assuming a constant number of active hypotheses over time, cf. also Fig. 2), however, may lead to a suboptimal solution, if the optimal one has been eliminated due to high local distances in early regions of the sequences. Nevertheless, in practice, beam search has been shown to be a successful heuristic.

For an effective beam search algorithm, one has to define a reasonable strategy, at what stage concurrent hypotheses should be compared. Most beam search algorithms in the HMM context handle this issue *frame-synchronous*, i.e., they compare and eliminate hypotheses along a line $\phi_{\boldsymbol{t}}(n) = \text{const}$ (each dashed vertical line in Fig. 3b). This strategy is well motivated by the fact that for the typical choice of transitions in HMMs such as $\mathbb{P} = \{(1,0),(1,1)\}$, all hypotheses have an equal amount of remaining transition steps to the endpoint $(N_{\boldsymbol{t}}, N_{\mathcal{R}})$ of the path.

However, for the symmetric transitions of (3) the pruning strategy also has to reflect the symmetry of the warping space. One solution and our choice is to compare and prune hypotheses *synchronous with respect to the matrix diagonals*, i.e., the lines $\phi_{\boldsymbol{t}}(n) + \phi_{\mathcal{R}}(n) = \text{const}$ (compare Fig. 3c).

The beam search strategy described above is applied for an evaluation of (2) *inside* the minimization of (11). In *frog on hand*, similar pruning strategies are employed *across* multiple evaluations of (2) in the minimization of (11). In this respect, hypotheses for the computation of the particular $\mathcal{R}^{lk}$-Viterbi matrix $D^*[\hat{d}](\boldsymbol{t},\mathcal{R}^{lk})$ are regularly compared with hypotheses for the Viterbi matrices of the other allograph models $D^*[\hat{d}](\boldsymbol{t},\mathcal{R}^{l'k'})$. If $\mathcal{R}^{lk}$ is unlikely to be the optimal solution in that context, the computation of $D^*[\hat{d}](\boldsymbol{t},\mathcal{R}^{lk})$ is aborted.

3.2 CSDTW Training

As motivated previously, CSDTW aims to combine two complementary strategies in order to cope with data variations. On a higher level, distinct writing styles are explicitly separated into subclasses. On a lower level, each of these writing styles is statistically modeled. A solution for these two issues is incorporated in the CSDTW training. To depict the idea beforehand in a few words, the first issue is treated by hierarchical cluster analysis, the second by maximum-likelihood (ML) parameter estimation. A thorough description will be given in the following.

As CSDTW being a generative approach, the training can be performed for each class independently. Given a set of data examples, provided with the corresponding character class labels, CSDTW training gives solutions to the following problems:

1. Generate allograph clusters $\mathbb{C}^{lk}, k = 1, \ldots K_l$ of the training set of class l. Since neither the data is labeled with cluster memberships, nor we want to label clusters by hand, this part corresponds to an unsupervised learning task.
2. Estimate a CSDTW reference model $\mathcal{R}^{lk}$ for each cluster $\mathbb{C}^{lk}$.

3.2.1 Generation of Allograph Clusters

According to Theodoridis and Koutroumbas [36, chapters 11-16], the term *clustering* describes the task of "'revealing' the organization of patterns into 'sensible' clusters (or groups)." A specification of the term "sensible" is highly dependent on factors like the underlying *pattern dissimilarity measure*, the *clustering criterion*, and the *clustering algorithm*, among others. Since these factors are application dependent, it remains to the clustering designer to define them properly with respect to prior knowledge about the problem.

In CSDTW, a cluster is modeled by unimodal probability densities. In this respect, it is favorable to assume compact clusters rather than elongated or shell-shaped ones. Further, since the number of clusters may be different for each class and not known a priori, a flexible treatment of the clustering granularity is desired. An *agglomerative hierarchical* clustering fulfills these requirements and will briefly be reviewed.

Agglomerative hierarchical clustering. In the context of handwriting, a *clustering* $\mathtt{C}^l$ defines a partitioning of a set of handwritten symbols $\mathbb{X}^l = \{\boldsymbol{x}^{l1}, \ldots, \boldsymbol{x}^{lM_l}\}$, i.e., all training examples of class l, into a set $\{\mathbb{C}^{l1}, \ldots, \mathbb{C}^{lK_l}\}$ of nonempty, pairwise disjoint sets, such that $\bigcup_{k=1}^{K_l} \mathbb{C}^{lk} = \mathbb{X}^l$. Since the clustering is computed independently for each class, we will omit the class index l in favor of a simpler notation in this section.

A *hierarchical clustering* algorithm produces a hierarchy of *nested* clusterings $[\mathtt{C}_1, \ldots, \mathtt{C}_M]$. The term "nested" defines the property that each cluster in $\mathtt{C}_{m-1}$ is a subset of a cluster in $\mathtt{C}_m$ and at least one of these subsets is a proper one. The hierarchy is generated iteratively in M steps, a clustering at

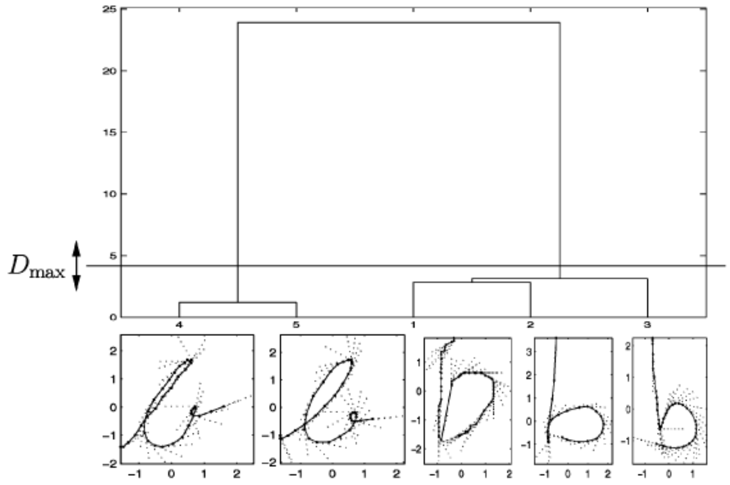


Fig. 4. A dissimilarity dendrogram of a clustering of five characters "b". Each leaf in the tree represents the below positioned character pattern; a node denotes a merge step of two particular clusters in Algorithm 1. The level of the node on the ordinate corresponds to the cluster dissimilarity $D(\mathbb{C}^k, \mathbb{C}^{k'})$, when merging. A specific value for $D_{\max}$ determines the granularity of the final clustering. The choice as is made in the figure will generate two clusters. For an interpretation of the character features cf. Fig. 1b.

step m is obtained from the clustering produced at the previous step $m-1$.

An *agglomerative hierarchical clustering* algorithm starts the iteration with a fine granularity of M clusters and ends with one cluster. Given a dissimilarity function $D(\mathbb{C}^k, \mathbb{C}^{k'})$ of two clusters, it uses the following general algorithm:

Algorithm 1: Hierarchical Agglomerative Clustering

```
1. Initialize the clustering $\mathtt{C}_1 = \{\{x^1\}, \ldots, \{x^M\}\}$
2. For $m = 2, \ldots, M$:
   obtain the new clustering $\mathtt{C}_m$ by merging clusters $\mathbb{C}^{k_{\min}}$
   and $\mathbb{C}^{k'_{\min}}$ of $\mathtt{C}_{m-1}$, where $(k_{\min}, k'_{\min}) = \mathrm{argmin}_{(k,k'),k\neq k'}$
   $D(\mathbb{C}^k, \mathbb{C}^{k'})$
```

Point and cluster dissimilarities. In CSDTW, we aim to reveal clusters based on a distance, which is consistent with the distance measure in classification. As argued in Section 3.1.4, the DTW Viterbi distance $D^*[\tilde{d}](\boldsymbol{x}^k, \boldsymbol{x}^{k'})$ satisfies this claim and, thus, is our favorable choice for the *point dissimilarity* in the cluster space,

$$D\left(\boldsymbol{x}^k, \boldsymbol{x}^{k'}\right) = D^*[\tilde{d}]\left(\boldsymbol{x}^k, \boldsymbol{x}^{k'}\right). \tag{12}$$

One possibility to define the *cluster dissimilarity* function $D(\mathbb{C}^k, \mathbb{C}^{k'})$ is to use the dissimilarity of its elements $D(\boldsymbol{x}^k, \boldsymbol{x}^{k'})$. More specifically, Theodoridis and Koutroumbas [36, chapter 11] suggest several particular combination methods like *min*, *max*, or *average dissimilarity*, however, the *average dissimilarity*

$$D\left(\mathbb{C}^k, \mathbb{C}^{k'}\right) = \frac{1}{O^k \cdot O^{k'}} \sum_{\boldsymbol{x}^k \in \mathbb{C}^k} \sum_{\boldsymbol{x}^{k'} \in \mathbb{C}^{k'}} D\left(\boldsymbol{x}^k, \boldsymbol{x}^{k'}\right), \tag{13}$$

gives a favor to compact clusters and achieved best recognition results in our experiments. Here, O^k denotes the cardinality of $\mathbb{C}^k$.

Cluster representatives. In many cases, it is useful to define a cluster representative. As will be argued in Section 3.2.2, CSDTW uses cluster representatives for initialization of the iterative parameter estimation.

In general, a cluster representative can be a point in the cluster space, a hyperplane or a hyperspherical representative. A suitable choice for compact clusters are *point representatives* $\tilde{\boldsymbol{x}}^k$. Theodoridis and Koutroumbas [36] describe different criteria for selecting point representatives; however, favor the *median center* for the case that the dissimilarity of two points is not a metric. In fact, this is true for (12). The median center $\tilde{\boldsymbol{x}}^k \in \mathbb{C}^k$ is defined by the property that it is the element with the smallest median distance with respect to the remaining cluster elements, i.e.,

$$\underset{\boldsymbol{x} \in \mathbb{C}^k, \boldsymbol{x} \neq \tilde{\boldsymbol{x}}^k}{\mathrm{med}} \left(D(\tilde{\boldsymbol{x}}^k, \boldsymbol{x})\right) \leq \underset{\boldsymbol{x} \in \mathbb{C}^k, \boldsymbol{x} \neq \boldsymbol{x}'}{\mathrm{med}} \left(D(\boldsymbol{x}', \boldsymbol{x})\right), \forall \boldsymbol{x}' \in \mathbb{C}^k. \tag{14}$$

Number of clusters. The result of an agglomerative hierarchical clustering algorithm can be visualized by a binary tree, a so-called *dissimilarity dendrogram*. Fig. 4 shows an example of a dissimilarity dendrogram for the clustering of five lower case characters "b".

A closer look at Fig. 4 reveals a strategy to determine a proper number of clusters. Instead of iterating all M steps in Algorithm 1, the merging can be stopped, when the cluster dissimilarity exceeds a threshold $D_{\max}$, i.e., $D(\mathbb{C}^{k_{\min}}, \mathbb{C}^{k'_{\min}}) > D_{\max}$. Figuratively, this approach cuts the dissimilarity dendrogram at the height $D_{\max}$ and uses the clustering assignment obtained from below that level.

Pruning clusters. We experienced that it is beneficial to eliminate clusters, the cardinality of which is smaller than a threshold $O_{\min}$. This benefit presumably arises from the following two reasons: First, small clusters are likely to be produced by data outliers. In the UNIPEN database, outliers are quite often present due to noisy and mislabeled data. Second, the statistical parameter estimation, as will be introduced in Section 3.2.2, is not robust for small clusters and should be avoided for those.

3.2.2 Estimation of Statistical Model Parameters

Each generated cluster is now to be modeled statistically. For the sake of completeness, we switch back to the previous nomenclature and again include the class index l in the cluster variables. In this notation, $\mathbb{C}^{lk}$ denotes the kth cluster of class l.

In the spirit of Section 3.1.2, $\mathbb{C}^{lk}$ is to be modeled by $\mathcal{R}^{lk} = [\mathbf{R}_1^{lk}, \ldots, \mathbf{R}_{N_{\mathcal{R}^{lk}}}^{lk}]$. Hence, the parameters to be

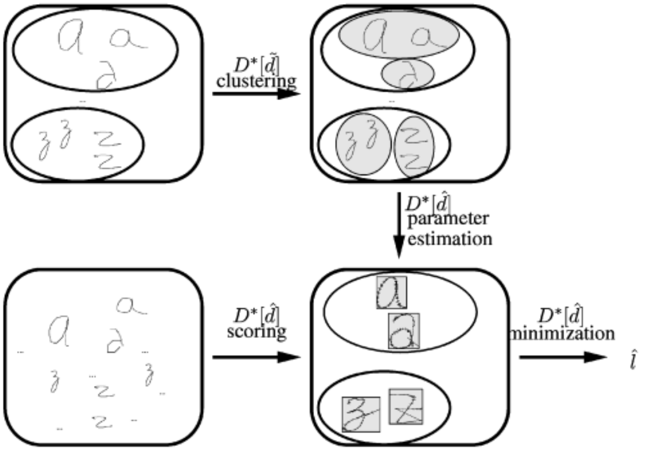


Fig. 5. A summary of the CSDTW training and classification. Training comprises clustering and parameter estimation, classification comprises scoring and minimizing. The clustering uses $D^*[\tilde{d}]$ as the underlying distance function, parameter estimation and scoring use $D^*[\hat{d}]$. Notably, $D^*[\tilde{d}]$ and $D^*[\hat{d}]$ are closely related. In this respect, CSDTW is a holistic combination of clustering and statistical sequence modeling.

estimated are: the mean $\boldsymbol{\mu}_j^{lk}$, the covariances Σ_j^{lk}, and the transition probabilities $\alpha_j^{lk}(\Delta\phi)$.

A common solution for the estimation problem is to pursue the maximum likelihood (ML) approach. ML seeks parameters $\mathcal{R}^{lk}$ that maximize the objective $\mathcal{L}(\mathcal{R}^{lk}) = \sum_{\boldsymbol{x}\in\mathbb{C}^{lk}} \ln p(\boldsymbol{x}|\mathcal{R}^{lk})$ [28], [32], [36].

This task already has been tackled in the HMM context and we can benefit from approved optimization algorithms. For its simplicity and speed, we use the decision directed *Viterbi training* (also known as *segmental K-means algorithm*) [18], [32], which instead of $\mathcal{L}(\mathcal{R}^{lk})$ maximizes the path-optimized variant $\mathcal{L}^*(\mathcal{R}^{lk}) = \sum_{\boldsymbol{x}\in\mathbb{C}^{lk}} \ln p(\boldsymbol{x}, \boldsymbol{\Phi}^*_{\boldsymbol{x},\mathcal{R}^{lk}}|\mathcal{R}^{lk})$. This approach gives estimates for the model parameters in an iteration of two basic steps.

Given an initial guess for $\mathcal{R}^{lk}$, step *one* computes the Viterbi alignments $\boldsymbol{\Phi}^*_{\boldsymbol{x},\mathcal{R}^{lk}}$ for $\boldsymbol{x} \in \mathbb{C}^{lk}$. In step *two*, an ML parameter reestimation of $\boldsymbol{\mu}_j^{lk}$ and Σ_j^{lk} is performed, based on all sample points that have been aligned to *j* in step *one*. The reestimated parameters then define a new model and the iteration is repeated. $\alpha_j^{lk}(\Delta\phi)$ is estimated in a similar fashion.

While the iteration is well-defined, theoretically founded, and prosperous, an open question in the context of the parameter estimation affects the problem of topology selection and initializing $\mathcal{R}^{lk}$.

The problem of topology selection is important since the number of model states, i.e., $N_{\mathcal{R}^{lk}}$ is not included in the ML optimization and has to be specified manually beforehand. On the other hand, the problem of initialization is important particularly with regard to the fact that the iterative training gives a *local* optimum to the ML criterion. The quantities among $\mathcal{R}^{lk}$, that are particularly sensitive to a proper initialization, are the parameters specifying the PDF [28, Chapter 6.12], i.e., $\boldsymbol{\mu}_j^{lk}$ and Σ_j^{lk} in our context.

CSDTW solves this problem by a unique, data-driven procedure. As the clustering was designed for generating compact clusters and point cluster representatives, the latter, in particular the median centers $\tilde{\boldsymbol{x}}^{lk} = [\tilde{\mathbf{x}}_1^{lk}, \ldots, \tilde{\mathbf{x}}_{N_{\tilde{\boldsymbol{x}}^{lk}}}^{lk}]$, are well-suited for topology selection and initialization. Thus, we initialize:

1. Mean $\boldsymbol{\mu}_j^{lk} = \tilde{\mathbf{x}}_j^{lk}, \forall j = 1, \ldots, N_{\tilde{\boldsymbol{x}}^{lk}}$,
2. Covariance $\Sigma_j^{lk} = \Sigma, \forall j = 1, \ldots, N_{\tilde{\boldsymbol{x}}^{lk}}$ (cf. Section 3.1.4), and
3. Transition probabilities $\alpha_j^{lk}(\Delta\phi) = 1/|\mathbb{P}|, \forall j = 1, \ldots, N_{\tilde{\boldsymbol{x}}^{lk}}, \forall \Delta\phi \in \mathbb{P}$.

Also in that regard, $N_{\mathcal{R}^{lk}}$ is implicitly fixed by this prescription to $N_{\tilde{\boldsymbol{x}}^{lk}}$. Thanks to the Sakoe-Chiba transitions $N_{\mathcal{R}^{lk}}$ can be arbitrary, contrary to HMM implementations without null-transitions.

Note that with the proposed initialization the SDTW Viterbi distances $D^*[\hat{d}](\boldsymbol{x}, \mathcal{R}^{lk})$, $\boldsymbol{x} \in \mathbb{C}^{lk}$ in the first Viterbi training iteration are equivalent to the DTW clustering dissimilarities $D^*[\tilde{d}](\boldsymbol{x}, \tilde{\boldsymbol{x}}^{lk})$, $\boldsymbol{x} \in \mathbb{C}^{lk}$, as it can be seen by a comparison of (5) and (7). This connection additionally reveals the attractive holistic property, regarding a seamless integration of clustering and statistical parameter estimation in CSDTW.

Finally, it should be mentioned that other training algorithms like the Baum-Welch parameter reestimation as well as discriminative approaches (maximum mutual information, MMI, minimum classification error, MCE) [17], [28], can also be employed instead of the Viterbi training, if desired.

3.3 CSDTW Concluded

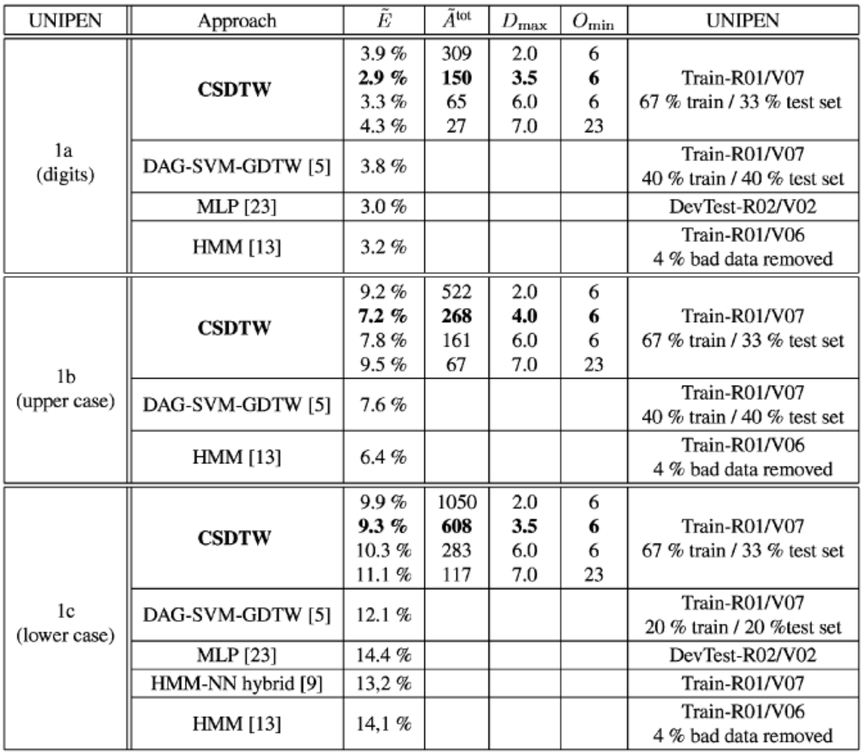
We have introduced CSDTW as an HMM-based classification approach that combines cluster analysis and statistical sequence modeling. On the one hand, it employs a specifically modified DTW Viterbi distance $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ in the cluster analysis. On the other hand, the SDTW Viterbi distance $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$ is used in the statistical parameter estimation and classification. Notably, $D^*[\tilde{d}](\boldsymbol{t}, \boldsymbol{r})$ is a special incarnation of $D^*[\hat{d}](\boldsymbol{t}, \mathcal{R})$. This aspect has two attractive effects. First, clustering and statistical modeling appear as a holistic combination, where cluster and classification space coincide. Clusters in the cluster space correspond to well-formed clusters in the classification space. Second, the result of the clustering can be consulted as a natural solution to the problem of topology selection and initializing the CSDTW reference models in the context of the iterative parameter estimation. In this respect, clustering and parameter estimation are seamlessly integrated into each other. Fig. 5 summarizes the training and classification issues graphically.

Another beneficial property of CSDTW is the following: With the clustering parameters $D_{\max}$ and $O_{\min}$ the classifier designer has direct influence on the granularity of the clustering. He can scale the classifier with respect to a wide range of cluster numbers and sizes, finding a compromise between the recognition accuracy and the computational time and memory requirements. A concrete example is the following: A large $D_{\max}$ implies that the role of the allograph models correspond to broader (maybe culturally conditioned) variations, while using a smaller $D_{\max}$ further reflects variations that may be due to different writer habits or the word context in that a character is written. Fig. 4 gives a supportive illustration for these ideas.

4 Experiments

In order to evaluate the performance of *frog on hand* with CSDTW and to compare with other recognition systems, a number of experiments have been carried out.

TABLE 1
Experiments on the UNIPEN Sections 1a/b/c (Indicated in the First Column)



UNIPEN	Approach	$\tilde{E}$	$\tilde{A}^{\rm tot}$	$D_{\max}$	$O_{\min}$	UNIPEN
1a (digits)	**CSDTW**	3.9 %	309	2.0	6	Train-R01/V07 67 % train / 33 % test set
		2.9 %	**150**	**3.5**	**6**	
		3.3 %	65	6.0	6	
		4.3 %	27	7.0	23	
	DAG-SVM-GDTW [5]	3.8 %				Train-R01/V07 40 % train / 40 % test set
	MLP [23]	3.0 %				DevTest-R02/V02
	HMM [13]	3.2 %				Train-R01/V06 4 % bad data removed
1b (upper case)	**CSDTW**	9.2 %	522	2.0	6	Train-R01/V07 67 % train / 33 % test set
		7.2 %	**268**	**4.0**	**6**	
		7.8 %	161	6.0	6	
		9.5 %	67	7.0	23	
	DAG-SVM-GDTW [5]	7.6 %				Train-R01/V07 40 % train / 40 % test set
	HMM [13]	6.4 %				Train-R01/V06 4 % bad data removed
1c (lower case)	**CSDTW**	9.9 %	1050	2.0	6	Train-R01/V07 67 % train / 33 % test set
		9.3 %	**608**	**3.5**	**6**	
		10.3 %	283	6.0	6	
		11.1 %	117	7.0	23	
	DAG-SVM-GDTW [5]	12.1 %				Train-R01/V07 20 % train / 20 %test set
	MLP [23]	14.4 %				DevTest-R02/V02
	HMM-NN hybrid [9]	13,2 %				Train-R01/V07
	HMM [13]	14,1 %				Train-R01/V06 4 % bad data removed

The second column denotes the classification approach used. Here, we also include other HWR approaches, as collected from the literature. The third column shows the mean error rate $\tilde{E}$ of five different data set partitionings. For CSDTW, it was computed for differently sized models, as indicated in the forth column by the average number of allographs $\tilde{A}^{\rm tot}$. The different model sizes were generated by varying $D_{\max}$ in the range $[2.0, \ldots, 7.0]$ and $O_{\min}$ in $[6, \ldots, 23]$ (see column 5 and 6). The CSDTW result with the lowest error rate is typed bold face. As some experiments were computed on different UNIPEN distributions, details are given in the last column.

4.1 Data

The experiments are based on sections 1a, 1b, and 1c (digits, upper, and lower case characters, respectively) of the UNIPEN [12] Train-R01/V07 database. For these sections, the data set size is ≈ 16K, 28K, and 61K characters, respectively The characters were randomly and disjointly divided into training and test sets of a ratio $2:1$. We evaluated in a *multiwriter* environment, i.e., the division was completely random and one writer was allowed to be present in both of the sets.[3] It should be noted that UNIPEN consists of very difficult data due to the variety of writers and noisy or mislabeled data. We used the database without cleaning in order to be as comparable as possible to other classification reports.

4.2 Results

For an effective evaluation, a few model parameters had to be set. A value for an initial, global covariance Σ (cf. Section 3.1.1) has been estimated on the basis of an ML estimation of previous models to $\Sigma = \mathrm{diag}(0.08, 0.05, 0.15)$.

Beside the error rate, a significant property of a CSDTW classification model is its size, i.e., the total number $A^{\rm tot}$ of generated allograph models. By varying $D_{\max}$ in the range $[2.0, \ldots, 7.0]$ (compare Fig. 4) and $O_{\min}$ in $[6, \ldots, 23]$, differently sized CSDTW classification models were generated. Table 1 summarizes mean classification error rates $\tilde{E}$ and model sizes $\tilde{A}^{\rm tot}$ for selected configurations, each of which is the average from five different data set partitionings.

From the table, it can be taken that an error minimum is given in all sections for $D_{\max} \approx 3.5$-4.0 and $O_{\min} = 6$, leading to model sizes of $\tilde{A}^{\rm tot} = 150$, 268, and 608 for the sections 1a/b/c, respectively. Notably, a further decrease of $D_{\max}$ (and, thus, an enlarging of the model size) does not improve the accuracy. On the other hand, the model size can significantly be reduced (by a factor ≈ 5) with $D_{\max} = 7.0$ and $O_{\min} = 23$. For this parameter configuration, the recognition accuracy decreases by absolute ≈ 2 percent, however, remains in an acceptable range.

For a comparison of the recognition accuracy in a global context, we collected other UNIPEN results from the literature and included them in Table 1. It must be stated that, though all experiments were computed on UNIPEN data, various reports used different character sets. Benchmarks were computed on miscellaneous versions and sizes of a UNIPEN database or some authors removed low quality/mislabeled characters, as indicated in the table's last column. Further, differences with respect to the multi and omniwriter testing environments aggravate a comparison, as argued above. In this respect, caution must be given when interpreting the rates. However, beyond this caution, a comparison of all results indicates that *frog on hand* with CSDTW classification achieves equivalent or superior rates with respect to all other approaches in the three categories. Especially for the

[3] Contrary, an *omniwriter* evaluation divides samples of one writer only in one of the sets but not in both, which is apparently a more difficult testing environment.

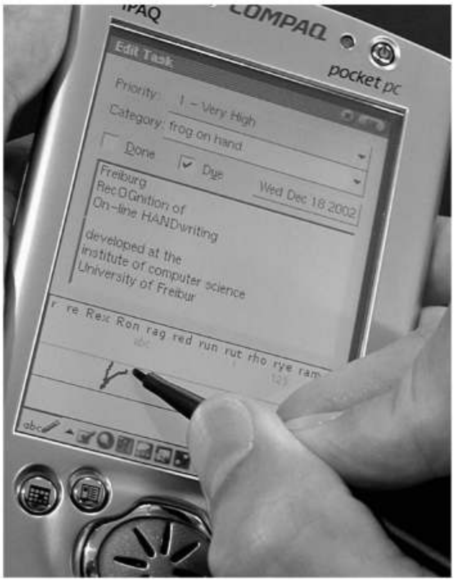


Fig. 6. *frog on hand* with CSDTW classification on a Linux Compaq iPAQ.

important lower case character section, *frog on hand* performs significantly better.

4.3 Complexity

An SDTW distance evaluation $D^*[\hat{d}](\boldsymbol{t},\mathcal{R})$ using beam search has the asymptotic complexity $C_{\mathrm{Time}}(D^*[\hat{d}](\boldsymbol{t},\mathcal{R})) = \mathcal{O}(\tilde{N}\cdot|\mathbb{P}|\cdot F^2)$ (cf. Section 3.1.5), with $\tilde{N}$ the average length of sequences $\mathcal{R}^{lk}$ and $\boldsymbol{t}$. F^2 ($=9$ in our case) corresponds to the multiplication with the covariance matrix. Experimentally, we measured $\tilde{N}=42$ and $\mathrm{Time}(D^*[\hat{d}](\boldsymbol{t},\mathcal{R})) \approx 0.0005\,\mathrm{sec}$ in our implementation on an AMD Athlon 1800MHz.

For a model size $A^{\mathrm{tot}}=600$, the classification time is $\mathrm{Time}(\mathrm{Classification}) \approx A^{\mathrm{tot}}\cdot\mathrm{Time}\Big(D^*[\hat{d}](\boldsymbol{t},\mathcal{R})\Big) = 600\cdot 0.0005$ $\mathrm{sec} = 0.3\,\mathrm{sec}$, across-model beam search (as explained in Section 3.1.5) further reduces $\mathrm{Time}(\mathrm{Classification})$ to $0.13\,\mathrm{sec}$.

The training complexity $C_{\mathrm{Time}}(\mathrm{Train})$ of CSDTW is dominated by the cluster analysis, which is asymptotically $C_{\mathrm{Time}}(\mathrm{Cluster}) = \mathcal{O}\big(L\cdot\big(\tilde{M}^2\cdot C_{\mathrm{Time}}\big(D^*[\tilde{d}](\boldsymbol{t},\boldsymbol{r})\big)+\tilde{M}^3\big)\big)$, with L the number of classes and $\tilde{M}$ the average number of training patterns in *one* class. The term $\tilde{M}^2\cdot C_{\mathrm{Time}}\big(D^*[\tilde{d}](\boldsymbol{t},\boldsymbol{r})\big)$ corresponds to the computation of the pairwise pattern dissimilarities, $\tilde{M}^3$ for the cluster linkage [36]. In the 1c experiments ($\tilde{M}\approx 1,500$), the training time was measured as $\mathrm{Time}(\mathrm{Train})$ $\approx 20\,\mathrm{h}$. Beneficially, training can easily be parallelized into L independent processes.

The memory complexity C_{Memory} basically consists of the storage of all CSDTW reference models. For $A^{\mathrm{tot}}=600$, $\mathrm{Memory}\approx A^{\mathrm{tot}}\cdot\tilde{N}\cdot(F+F^2+|\mathbb{P}|\cdot F)\cdot 4\,\mathrm{byte} = 600\cdot 42\cdot(3$ $+9+9)\cdot 4\,\mathrm{byte} = 2067\,\mathrm{Kbyte}$ without compression.

5 Application

In order to demonstrate *frog on hand*'s performance in a typical application, we have implemented the system using the CSDTW classification on a PDA, a Compaq iPAQ 3660. This device is equipped with a 203 MHz StrongARM processor, 16 MByte flash memory (corresponding to a PC's hard drive) and 64 MByte RAM. It is operated by a Linux kernel 2.4.18 from the *Familiar* distribution,[4] the graphical user interface is the *Open Palmtop Integrated Environment (OPIE)*,[5] which is a GPL licensed fork of Trolltech's *Qtopia*.[6]

Using the approach described in the sections above, we have developed an integrated character recognition plug-in that runs both inside Qtopia and OPIE.[7] The character label recognized by the plug-in is transferred to the active application (e.g., date book, address book, to-do list, etc.). Contrary to the implementation in our development environment, a fixed point data representation is used on the PDA, because the StrongARM processor lacks a built-in floating point arithmetic.

As known from commercial character recognition programs, the input region is divided into three zones, each zone corresponding to one of the UNIPEN 1a/b/c sections and a set $\Re = \{\mathcal{R}^{lk}\}_{l\in\{1,\ldots,L\},k\in\{1,\ldots,K_l\}}$ of CSDTW reference models. For the PDA, we were able to benefit from the scalability of the CSDTW approach and have chosen a rather small set of reference models with the cluster parameter configuration $D_{\max}=7.0$ and $O_{\min}=23$ (compare Table 1).

The classification of a character approximately takes 0.3 seconds in average on the iPAQ. The picture of Fig. 6 should give an idea of the environment.

6 Conclusion

In this contribution, we have described our writer independent online handwriting recognition system *frog on hand*, focusing on the HMM-based classification approach CSDTW. CSDTW is a scalable solution for the task of holistically combining clustering and statistical sequence modeling. In contrast to previous approaches, cluster analysis and statistical classification take place in the same feature space and use a closely related distance measure. This has attractive consequences. First, clusters found in the cluster space correspond to well-formed clusters in the classifier space. Second, the result of the clustering can be used to define the topology selection and initialization of the statistical sequence models in the context of the iterative statistical parameter estimation.

Using a hierarchical cluster analysis, CSDTW includes a method to scale the size of the classifier, finding a compromise between the recognition accuracy and the computation time and memory requirements of the underlying device.

Experiments have shown that *frog on hand* with CSDTW achieves lower error rates than other approaches on UNIPEN data, which is the standard benchmark in online HWR. Especially for lower case characters, where a large variety of writing styles is present, the presented system provides a major improvement over other HWR systems. We see three main reasons for the superior recognition accuracy. First, the powerful cluster analysis can adequately model variations in the writing style. It also includes a mechanism to detect and eliminate outliers in the training data set that may correspond to mislabeled data. Second, CSDTW employs a data-driven topology selection and initialization of the statistical warping framework. Due to the chosen local transitions, the statistical sequence models can be of arbitrary

[4] http://familiar.handhelds.org/.

[5] http://opie.handhelds.org/.

[6] http://www.trolltech.com/products/qtopia/.

[7] A binary package of the implementation is available from our WWW site http://lmb.informatik.uni-freiburg.de/people/bahlmann/frog.en.html.

length. Third, current studies indicate [2] that a special statistical modeling of the feature θ (the tangent slope angle), which takes into account the distinctive circular nature of θ, gives significant improvements in error rate compared to previous approaches.

It should be stated that CSDTW is a general concept that can advantageously be applied to other problems where data are a sequence of feature vectors (or a set of one-dimensionally sorted elements). These include, e.g., speech recognition [28], genome processing [14], robotics [22], etc.

We have demonstrated *frog on hand*'s efficiency and suitability for real word applications through the implementation on a Linux Compaq iPAQ PDA. Here, we could advantageously benefit from CSDTW's scalability.

We see interesting issues for additional research. In current work, we are extending the character recognition onto a word-level basis.

In order to speed up classification and to decrease the model size, a method for a subsampling of the allograph models would be an attractive area for further studies. In fact, Fig. 1c indicates that the number of sample points could be reduced without loosing much accuracy. Similar to the current model topology selection, the subsampling should preferably be solved automatically and data-driven.

The presented approach has assumed a feature space with compact, Gaussian-like clusters. However, clusters in many real-world problems often have a more complex form, e.g., they are elongated or shell-shaped. Clustering methods exist [40] that cope with this issue by mapping the input space with a so-called kernel. A future challenge is to integrate such a kernel formulation jointly into the clustering *and* the statistical, generative sequence modeling.

Recently, Vuori et al. [37] proposed methods for an unsupervised, writer *dependent* adaptation by the elimination of character prototypes from the reference set. This idea, originally developed for simple character templates, can straightforwardly be transferred to the CSDTW allograph models.

CSDTW is a generative classification approach. As generative and discriminative classifiers employ different philosophies and have different strengths and weaknesses, a combination of both types is a promising approach. A very successful discriminative classifier today is the support vector machine (SVM). However, common SVM techniques were developed for a feature space with a fixed dimension, whereas the vector sequences vary in length and are temporally distorted. In our recent research [5], we have demonstrated how this classifier can be applied to pattern sequences like online HWR data. This work serves as a basis for the above noted classifier combination.

Acknowledgments

The authors would like to thank Bernard Haasdonk, Klaus-Dieter Peschke, Kai Simon, and Cynthia Findlay from the Computer Science Department at the University of Freiburg for valuable discussions, comments, and corrections. Thanks also to Dirk Bockhorn who worked out valuable ideas, which are found in this paper. Further, they would like to thank the anonymous reviewers for their constructive comments.

References

[1] L.R. Bahl, F. Jelinek, and R.L. Mercer, "A Maximum Likelihood Approach to Continuous Speech Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 5, pp. 179-190, Mar. 1983.

[2] C. Bahlmann, "Directional Features in On-Line Handwriting Recognition," technical report, Albert-Ludwigs-Universität Freiburg, Institut für Informatik, http://lmb.informatik.uni-freiburg.de/, 2003.

[3] C. Bahlmann, "Advanced Sequence Classification Techniques Applied to On-Line Handwriting Recognition," PhD thesis, Albert-Ludwigs-Universität Freiburg, Institut für Informatik, to appear, http://lmb.informatik.uni-freiburg.de/, 2004.

[4] C. Bahlmann and H. Burkhardt, "Measuring HMM Similarity with the Bayes Probability of Error and Its Application to Online Handwriting Recognition," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 406-411, 2001.

[5] C. Bahlmann, B. Haasdonk, and H. Burkhardt, "On-Line Handwriting Recognition with Support Vector Machines—A Kernel Approach," *Proc. Eighth Int'l Workshop Frontiers in Handwriting Recognition,* pp. 49-54, 2002.

[6] S. Bercu and G. Lorette, "On-Line Handwritten Word Recognition: An Approach Based On Hidden Markov Models," *Proc. Third Int'l Workshop Frontiers in Handwriting Recognition,* pp. 385-390, 1993.

[7] S.D. Connell and A.K. Jain, "Learning Prototypes for On-Line Handwriting Digits," *Proc. 12th Int'l Conf. Pattern Recognition,* pp. 182-184, 1994.

[8] A. Dengel, R. Hoch, F. Hönes, T. Jäger, M. Malburg, and A. Weigel, "Techniques for Improving OCR Results," *Handbook of Character Recognition and Document Image Analysis,* H. Bunke and P.S.P. Wang, eds., pp. 227-258, World Scientific, 1997.

[9] N. Gauthier, T. Artières, B. Dorizzi, and P. Gallinari, "Strategies for Combining On-Line and Off-Line Information in an On-Line Handwriting Recognition System," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 412-416, 2001.

[10] W. Guerfali and R. Plamondon, "Normalizing and Restoring On-Line Handwriting," *Pattern Recognition,* vol. 16, no. 5, 1993.

[11] I. Guyon, P. Albrecht, Y. LeCun, J.S. Denker, and W. Hubbard, "Design of a Neural Network Character Recognizer for a Touch Terminal," *Pattern Recognition,* vol. 24, no. 2, pp. 105-119, 1991.

[12] I. Guyon, L.R.B. Schomaker, R. Plamondon, M. Liberman, and S. Janet, "UNIPEN Project of On-Line Data Exchange and Recognizer Benchmarks," *Proc. 12th Int'l Conf. Pattern Recognition,* pp. 29-33, 1994, http://www. unipen. org/.

[13] J. Hu, S.G. Lim, and M.K. Brown, "Writer Independent On-Line Handwriting Recognition Using an HMM Approach," *Pattern Recognition,* vol. 33, pp. 133-147, Jan. 2000.

[14] T. Jaakkola, M. Diekhans, and D. Haussler, "Using the Fisher Kernel Method to Detect Remote Protein Homologies," *Proc. Eighth Int'l Conf. Image Analysis and Processing,* 1999.

[15] S. Jäger, S. Manke, and A. Waibel, "NPEN++: An On-Line Handwriting Recognition System," *Proc. Seventh Int'l Workshop Frontiers in Handwriting Recognition,* pp. 249-260, 2000.

[16] F. Jelinek, *Statistical Methods for Speech Recognition.* MIT Press, 1998.

[17] B.H. Juang, W. Chou, and C.H. Lee, "Minimum Classification Error Rate Methods for Speech Recognition," *IEEE Trans. Speech & Audio Processing,* vol. 5, no. 3, pp. 257-265, May 1997.

[18] B.H. Juang and L.R. Rabiner, "The Segmental k-Means Algorithm for Estimating Parameters of Hidden Markov Models," *IEEE Trans. Acoustic Speech Signal Processing,* vol. 38, pp. 1639-1641, 1990.

[19] J.J. Lee, J. Kim, and J.H. Kim, "Data Driven Design of HMM Topology for On-Line Handwriting Recognition," *Proc. Seventh Int'l Workshop Frontiers in Handwriting Recognition,* pp. 239-248, 2000.

[20] S. Manke, M. Finke, and A. Waibel, "A Fast Search Technique for Large Vocabulary On-Line Handwriting Recognition," *Proc. Fifth Int'l Workshop Frontiers in Handwriting Recognition,* 1996.

[21] T. Oates, L. Firoiu, and P. Cohen, "Clustering Time Series with Hidden Markov Models and Dynamic Time Warping," *Proc. IJCAI-99 Workshop on Neural, Symbolic, and Reinforcement Learning Methods for Sequence Learning,* pp. 17-21, 1999.

[22] T. Oates, M.D. Schmill, and P. Cohen, "A Method for Clustering the Experiences of a Mobile Robot that Accords with Human Judgments," *Proc. 17th Nat'l Conf. Artificial Intelligence,* pp. 846-851, 2000.

[23] M. Parizeau, A. Lemieux, and C. Gagnè, "Character Recognition Experiments Using UNIPEN Data," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 481-485, 2001.

[24] M.P. Perrone and S.D. Connell, "K-Means Clustering for Hidden Markov Models," *Proc. Seventh Int'l Workshop Frontiers in Handwriting Recognition,* pp. 229-238, 2000.

[25] R. Plamondon, D. Lopresti, L.R.B. Schomaker, and R. Srihari, "On-Line Handwriting Recognition," *Encyclopedia of Electrical and Electronics Eng.,* J.G. Webster, ed., vol. 15, pp. 123-146, Wiley InterScience, 1999.

[26] R. Plamondon and S. N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 63-84, Jan. 2000.

[27] L. Prevost and M. Milgram, "Non-Supervised Determination of Allograph Sub-Classes for On-Line Omni-Scriptor Handwriting Recognition," *Proc. Fifth Int'l Conf. Document Analysis and Recognition,* 1999.

[28] L.R. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[29] G. Rigoll, A. Kosmala, and D. Willett, "A New Hybrid Approach to Large Vocabulary Cursive Handwriting Recognition," *Proc. 12th Int'l Conf. Pattern Recognition,* pp. 1512-1514, 1994.

[30] M. Schenkel, I. Guyon, and D. Henderson, "On-Line Cursive Script Recognition Using Time Delay Neural Networks and Hidden Markov Models," *Machine Vision and Applications 8,* R. Plamondon, ed., pp. 215-223, Springer Verlag, 1995.

[31] L.R.B. Schomaker, "Using Stroke- or Character-Based Self-Organizing Maps in the Recognition of On-Line, Connected Cursive Script," *Pattern Recognition,* vol. 26, no. 3, pp. 443-450, 1993.

[32] E.G. Schukat-Talamazzini, *Automatische Spracherkennung—Grundlagen, Statistische Modelle und Effiziente Algorithmen.* Friedr. Vieweg & Sohn, 1995.

[33] G. Seni, R.K. Srihari, and N. Nasrabadi, "Large Vocabulary Recognition of On-Line Handwritten Cursive Words," *IEEE Trans. Pattern Analysis and Machine Intelligence,* pp. 757-762, 1996.

[34] K. Simon, "Erkennung von Handgeschriebenen Wörtern mit CSDTW," Master's thesis, Albert-Ludwigs-Universität Freiburg, Institut für Informatik, http://lmb.informatik.uni-freiburg.de/, 2003.

[35] P. Smyth, "Clustering Sequences with Hidden Markov Models," *Advances in Neural Information Processing Systems 9,* pp. 648-654, 1997.

[36] S. Theodoridis and K. Koutroumbas, *Pattern Recognition.* Academic Press, 1999.

[37] V. Vuori, J. Laaksonen, E. Oja, and J. Kangas, "Experiments with Adaptation Strategies for a Prototype-Based Recognition System for Isolated Handwritten Characters," *Int'l J. Document Analysis and Recognition,* vol. 3, no. 2, pp. 150-159, 2001.

[38] L.G. Vuurpijl and L.R.B. Schomaker, "Finding Structure in Diversity: A Hierarchical Clustering Method for the Categorization of Allographs in Handwriting," *Proc. Fourth Int'l Conf. Document Analysis and Recognition,* pp. 387-393, 1997.

[39] G.W. Wilfong, F. Sinden, and L. Ruedisueli, "On-Line Recognition of Handwritten Symbols," *IEEE Trans. Pattern Analysis and Machine Intelligence,* pp. 935-940, 1996.

[40] R. Zhang, A.I. Rudnicky, "A Large Scale Clustering Scheme for Kernel k-Means," *Proc. 16th Int'l Conf. Pattern Recognition,* 2002.

Claus Bahlmann received the BS and MS degrees in computer science from the University of Bielefeld, Germany in 1997. He is currently finishing his doctoral dissertation that is concerned with new types of generative and discriminative classification for online handwriting recognition. He joined the Computer Science Department, University of Freiburg, Germany in 1998 as a research associate. In 2002, his work was awarded the "Best Paper Presentation" prize at the IWFHR 2002 conference in Niagara-on-the-Lake, Canada. His main research interests are handwriting recognition, pattern recognition, machine learning, and computer vision.

Hans Burkhardt received the Dipl.-Ing. degree in electrical engineering in 1969, Dr.-Ing. degree in 1974, and the Venia Legendi in 1979 from the University of Karlsruhe, Germany. From 1969, he was research assistant and, in 1975, he became a lecturer at the University of Karlsruhe. During 1980-1981, he had a scientific fellowship at the IBM Research Laboratory, San Jose, California. In 1981, he became a professor for control and signal theory at the University of Karlsruhe. During 1985-1996, he was a full professor at the Technical University of Hamburg and director of an Institute in the Computer Science Department and, additionally, a scientific advisor between 1990 and 1996 for the Microelectronic Application Center (MAZ) in Hamburg. Since 1997, he has been a full professor at the Computer Science Department of the University of Freiburg, director of an Institute for Pattern Recognition and Image Processing and, currently, Deputy Dean of the Faculty for Applied Sciences. Since 2000, he has been president of the German Association for Pattern Recognition (DAGM). He is a member of the "Academy of Sciences and Humanities, Heidelberg" and a fellow of the International Association for Pattern Recognition (IAPR). He is currently on sabbatical leave at the National ICT Australia (NICTA), Department of Systems Engineering (RSISE) at the Australian National University (ANU), Canberra ACT 0200, Australia. He has published more than 150 papers and given more than 200 lectures. He is a consultant for several national and international institutions, e.g., the German Science Foundation (DFG), the European Commission and different international organizations and journals. In 1998, he was chair of the European Conference on Computer Vision (ECCV). Experience: Invariants in pattern recognition, optimal image restoration methods, motion estimation algorithms, parallel algorithms in image processing and pattern recognition, image analysis, and vision guided control of combustion processes. He is a member of the IEEE and the IEEE Computer Society.