publ. in Proc. of the 8th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR), pp. 49–54, 2002.
ftp://ftp.informatik.uni-freiburg.de/papers/lmb/ba_ha_bu_iwfhr02.pdf

On-line Handwriting Recognition with Support Vector Machines— A Kernel Approach

Claus Bahlmann, Bernard Haasdonk and Hans Burkhardt
Computer Science Department
Albert-Ludwigs-University Freiburg
79110 Freiburg, Germany
{bahlmann,haasdonk,burkhardt}@informatik.uni-freiburg.de

Abstract

In this contribution we describe a novel classification approach for on-line handwriting recognition. The technique combines dynamic time warping (DTW) and support vector machines (SVMs) by establishing a new SVM kernel. We call this kernel Gaussian DTW (GDTW) kernel. *This kernel approach has a main advantage over common HMM techniques. It does not assume a model for the generative class conditional densities. Instead, it directly addresses the problem of discrimination by creating class boundaries and thus is less sensitive to modeling assumptions. By incorporating DTW in the kernel function, general classification problems with variable-sized sequential data can be handled. In this respect the proposed method can be straightforwardly applied to all classification problems, where DTW gives a reasonable distance measure, e.g. speech recognition or genome processing. We show experiments with this kernel approach on the UNIPEN handwriting data, achieving results comparable to an HMM-based technique.*

1. Introduction

The utilization of support vector machine (SVM) [2, 4] classifiers has gained immense popularity in the last years. SVMs have achieved excellent recognition results in various pattern recognition applications [4]. Also in off-line optical character recognition (OCR) they have been shown to be comparable or even superior to the standard techniques like Bayesian classifiers or multilayer perceptrons [5]. SVMs are *discriminative* classifiers based on Vapnik's *structural risk minimization principle*. They can implement flexible decision boundaries in high dimensional feature spaces. The implicit regularization of the classifier's complexity avoids overfitting and mostly this leads to good generalizations. Some further properties are commonly seen as reasons for the success of SVMs in real-world problems: the optimality of the training result is guaranteed, fast training algorithms exist and little a-priori knowledge is required, i.e. only a labeled training set.

For the solution of on-line handwriting recognition (HWR) tasks researchers presently use classification methods which are based on a Bayesian *generative* approach: hidden Markov models (HMMs) model a sequence of class conditional densities based on (and thus restricted to) a certain function class. A discriminant function is obtained in a second step using Bayes' rule. Indeed HMMs have proven to deal very well with the complex on-line handwriting data structure. This is usually a variable-size *sequence* of feature vectors that may have been distorted in particular ways, each vector computed from sampled coordinates of the pen tip curve.

While the generative approach is indeed optimal if the underlying models are accurate, it can perform poorly if this assumption is not fulfilled. In most practical applications, also in HWR, the latter is often the case. In these situations discriminative approaches which do not aim to estimate class conditional densities but directly address the discrimination by creating class boundaries are a promising choice. As noted above, SVMs belong to this category of classifiers.

However, so far no on-line HWR system using SVMs is known to the authors. In this respect this contribution is the first one incorporating SVMs into on-line HWR. The reason why SVMs have not been used in the past can be seen in the data structure mentioned above: common SVM techniques were developed for a feature space with a fixed dimension, whereas the on-line handwriting sequences vary in length and are temporally distorted. An ad-hoc solution to overcome this incompatibility—like a linear scaling of the writing to a fixed number of samples—is not promising to outperform standard HMM techniques, since it cannot deal with the nonlinear, temporal variations in the data. In our view a successful approach should both embed the discriminative power of SVMs as well as the flexibility in coping with temporal distortions.

A starting point for linking SVMs with sequential data is the so-called *kernel*, as will be shown. Some work has been done in other research areas dealing with kernels for sequential data. Jaakkola et al. [10] developed an SVM kernel in their application of protein homology detection and refer to it as *Fisher kernel*. Watkins [19] describes several explicit kernels for sequential data and shows that the *joint probability of two sequences* according to a *pair HMM* is a proper SVM kernel under certain conditions.

Since the kernels mentioned above are still based on an estimation of generative parameters, we propose an alternative approach which is less complex and presumes less model knowledge: the *Gaussian dynamic time warping kernel (GDTW)*. We shall start with a short review of dynamic time warping (DTW) and SVMs in the following section. Section 3 then introduces the GDTW kernel. Experimental results with this GDTW kernel on the UNIPEN [7] data and a comparison to UNIPEN results of other recognition techniques are presented in section 4. Section 5 provides a conclusion of this contribution.

2. Background

2.1. Dynamic time warping

In DTW [15] a distance $D(\mathcal{T},\mathcal{R})$ from two vector sequences $\mathcal{T} = (\mathbf{t}_1,\ldots,\mathbf{t}_{N_\mathcal{T}})$ and $\mathcal{R} = (\mathbf{r}_1,\ldots,\mathbf{r}_{N_\mathcal{R}})$ is determined. In on-line HWR the vectors $\mathbf{t}_i \in I\!R^F$ and $\mathbf{r}_j \in I\!R^F$ are usually computed from the local neighborhood of the i-th respective j-th sampled point of the pen tip curve. See section 4.2 for the authors' choice of $\mathbf{t}_i$ and $\mathbf{r}_j$.

Given a so-called *alignment* (or *warping*) *path* $\boldsymbol{\phi} = (\phi(1),\ldots,\phi(N))$ with $\phi(n) = (\phi_\mathcal{T}(n),\phi_\mathcal{R}(n)) \in \{1,\ldots,N_\mathcal{T}\} \times \{1,\ldots,N_\mathcal{R}\}$($\boldsymbol{\phi}$ is introduced to align corresponding regions in the sequences $\mathcal{T}$ and $\mathcal{R}$; see the textbook of Rabiner and Juang [15, Chapter4.7] for further details) and a local distance measure d, e.g. $d(\mathbf{t}_i,\mathbf{r}_j) = \|\mathbf{t}_i - \mathbf{r}_j\|^2$, we define the *alignment distance* $D_{\boldsymbol{\phi}}$ as the mean distance along a particular alignment path $\boldsymbol{\phi}$

$$D_{\boldsymbol{\phi}}(\mathcal{T},\mathcal{R}) = \frac{1}{N}\sum_{n=1}^{N} d\left(\mathbf{t}_{\phi_\mathcal{T}(n)},\mathbf{r}_{\phi_\mathcal{R}(n)}\right). \quad (1)$$

The *DTW distance* (or *Viterbi distance*) $D(\mathcal{T},\mathcal{R})$ is defined as the alignment distance (1) along the *Viterbi path* $\boldsymbol{\phi}^*$

$$D(\mathcal{T},\mathcal{R}) = D_{\boldsymbol{\phi}^*}(\mathcal{T},\mathcal{R}) = \min_{\boldsymbol{\phi}}\{D_{\boldsymbol{\phi}}(\mathcal{T},\mathcal{R})\}. \quad (2)$$

It is convenient to model $\boldsymbol{\phi}$ as a sequence of local transitions. We use the ones which are known as *Sakoe-Chiba transitions* in the literature [15]. These only allow forward steps of size 1 in $\mathcal{T}$, $\mathcal{R}$ or in both of them, i.e. $\phi(n+1) - \phi(n)$ equals $(1,0)$, $(0,1)$ or $(1,1)$.

Usual dynamic programming and beam search strategies [11] are applied to reduce the computational complexity when minimizing (2).

The DTW technique itself in combination with a minimum distance classifier [17, 18] as well as the incorporation of statistical knowledge to this concept [1] have been successfully applied to handwriting recognition.

2.2. Support vector classification

Here, we provide a brief introduction to support vector classification. For more details and geometrical interpretations please refer to the standard literature, e.g. by Burges [2] or Cristianini and Shawe-Taylor [4].

Consider a two-class classification problem and a set of training vectors $\{P_i\}_{i=1,\ldots,M}$ with corresponding binary labels $S_i = 1$ for the "positive" and $S_i = -1$ for the "negative" class. In classification an SVM assigns a label $\hat{S}$ to a test vector T by evaluating

$$f(T) = \sum_i \alpha_i S_i K(T,P_i) + b \quad \text{and} \quad \hat{S} = \operatorname{sign}(f(T)). \quad (3)$$

The *weights* α_i and the *bias* b are SVM parameters and adopted during training by maximizing

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j S_i S_j K(P_i,P_j) \quad (4)$$

under the constraints

$$0 \le \alpha_i \le C \quad \text{and} \quad \sum_i \alpha_i S_i = 0 \quad (5)$$

with C a positive constant weighting the influence of training errors. $K(\cdot,\cdot)$ is the *kernel* of the SVM. A solution for the α_i implies a value for b. The SVM framework gives some flexibility in designing an appropriate kernel for the underlying application. Many implementations of kernels have been proposed so far, one popular example is the *Gaussian kernel*

$$K(P_i,P_j) = \exp\left(-\gamma\|P_i - P_j\|^2\right). \quad (6)$$

If $K(\cdot,\cdot)$ is positive definite, (4)–(5) is a convex quadratic optimization problem, for which the convergence towards the global optimum can be guaranteed. However, obtaining this solution for real-world problems can be quite demanding and requires sophisticated optimization algorithms like *chunking*, *decomposition* or *sequential minimal optimization* [4].

Usually $\alpha_i = 0$ for the majority of i and thus the summation in (3) is limited to a subset of the P_i, which therefore is called the set of *support vectors*.

Extensions of the binary classification to the multi-class situation are suggested in several approaches [2, 13].

3. Gaussian dynamic time warping kernel

As indicated in the introduction, when dealing with sequential on-line handwriting data we cannot simply employ the basic SVM framework given by (3)–(6). Different feature vector sequences $\mathcal{P}_i, \mathcal{P}_j$ and $\mathcal{T}$ cannot be embedded in the same vector space in general, as the necessary dimensions differ.

However, an important property of (3)–(4) is that the vectors P_i, P_j and T appear only in form of kernel evaluations. Thus our objective, when adopting SVMs to sequential handwriting data, can be to state a kernel definition suitable to the particular properties of the sequential data.

An obvious modification of (6) is to replace the squared Euclidean distance $\|P_i - P_j\|^2$ with the equivalent when dealing with temporally distorted, sequential signals—the DTW distance $D(\mathcal{P}_i, \mathcal{P}_j)$ described above. For two equally sized sequences and a linear alignment $\phi(n) = (n, n)$ even the equation $N \cdot D(\mathcal{P}_i, \mathcal{P}_j) = \|\mathcal{P}_i - \mathcal{P}_j\|^2$ holds.

We therefore apply this modification and define the *Gaussian DTW (GDTW) kernel* for sequential data by

$$K(\mathcal{P}_i, \mathcal{P}_j) = \exp\left(-\gamma D(\mathcal{P}_i, \mathcal{P}_j)\right). \tag{7}$$

In the following we will abbreviate an SVM classifier with a GDTW kernel as SVM-GDTW. Figure 1 illustrates the GDTW kernel with some examples.

A theoretical relevant property of (7) should be mentioned: as the DTW distance is not a metric (invalid triangle inequality in many cases), one could fear that the resulting kernel lacks some necessary properties, e.g. the positive definiteness (pd) noted in section 2.2. This would imply that the solution of the optimization algorithm is not guaranteed to be the global optimum. In fact general pd cannot be proven for (7), as simple counterexamples can be found. Nevertheless such kernels can produce good results like in our case and others [5, 8]. Recalling the fact that positive definite kernels are characterized by the property of generating kernel matrices $K_{ij} = K(\mathcal{P}_i, \mathcal{P}_j)$ with solely nonnegative eigenvalues λ_i, some reasons for the good recognition rates may be the following: Firstly, the SMO algorithm is operating on 2×2 kernel matrices and such matrices provable always have nonnegative λ_i. The SMO algorithm therefore is expected to converge although no optimality can be guaranteed. Secondly, larger kernel matrices generated

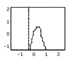
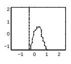
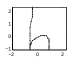
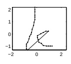
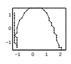
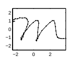
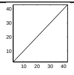
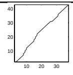
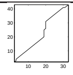
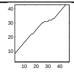
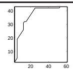
$\mathcal{T}$	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_3$	$\mathcal{P}_4$	$\mathcal{P}_5$
					
ϕ^*					
$D(\mathcal{T}, \mathcal{P}_j)$	0	0.20	0.71	0.99	10.04
$K(\mathcal{T}, \mathcal{P}_j)$	1	0.70	0.28	0.17	0.00

Figure 1. The Gaussian dynamic time warping (GDTW) kernel: Character patterns $\mathcal{T}$ (class "h") and $\mathcal{P}_j$, $j = 1, \ldots, 5$ (classes "h", "h", "k", "n", "m") are illustrated by the features $\tilde{x}$ and $\tilde{y}$ (see section 4.2 for the definition). The values for the DTW distance D and the GDTW kernel evaluation K for $\gamma = 1.8$ are provided in the third and fourth row, respectively. The values show the obvious fact that similar patterns give small values for D and large for K. In the second row the Viterbi path ϕ^* is illustrated: The sketched line traverses all aligned point pairs $\left(\phi^*_{\mathcal{T}}(n), \phi^*_{\mathcal{P}_j}(n)\right)$, $n = 1, \ldots, N$ in the $N_{\mathcal{T}} \times N_{\mathcal{P}_j}$ matrix.

by randomly chosen characters $\mathcal{P}_i, \mathcal{P}_j$ from the UNIPEN database turn out to violate the pd only weakly: For matrices of sizes up to 40×40 all λ_i were experimentally measured to be nonnegative. For larger matrices the sporadic missing pd was due to only a few negative eigenvalues with small absolute values compared to the other eigenvalues.

4. Experiments

4.1. Data

The experiments are based on the 1a, 1b and 1c section (digits, upper and lower case characters, respectively) of the UNIPEN [7] Train-R01/V07 database. For these sections the data set size is $\approx$ 16K, 28K and 61K, respectively. Examples of UNIPEN data were shown in figure 1. Training and test set were taken disjointly. It should be stated 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. Feature selection

We model each vector $\mathbf{t}_i$ of a sequence $\mathcal{T} = (\mathbf{t}_1, \ldots, \mathbf{t}_{N_{\mathcal{T}}})$ by three local features at the i-th sampled point (x_i, y_i): $\mathbf{t}_i = (\tilde{x}_i, \tilde{y}_i, \theta_i)^T$. The quantities $\tilde{x}_i = \frac{x_i - \mu_x}{\sigma_y}$ and $\tilde{y}_i = \frac{y_i - \mu_y}{\sigma_y}$ are the pen coordinates normalized by the mean (μ_x, μ_y) and y-deviation σ_y of all character's sample points. The feature θ_i is the tangent slope angle at point i, approximated by $\theta_i = \text{ang}\left((x_{i+1} - x_{i-1}) + \sqrt{-1} \cdot (y_{i+1} - y_{i-1})\right)$ with $\sqrt{-1}$ the imaginary unit and "ang" the complex angle function. Since θ_i is a circular measure, there is some special treatment necessary when computing the local distance $d\left(\mathbf{t}_i, \mathbf{r}_j\right) = \|\mathbf{t}_i - \mathbf{r}_j\|^2$. Instead of the usual difference $\Delta\theta = (\theta_1 - \theta_2)$ we use the circular difference $\Delta_{\text{mod}}\theta = (\theta_1 - \theta_2) \bmod 2\pi$ with $\Delta_{\text{mod}}\theta \in (-\pi, \pi]$.

No pre-processing, such as re-sampling of the writing or reference line detection, is applied in this case. Each pattern is typically represented by about 10–80 sample points.

4.3. Two-class experiments

We have trained the SVM-GDTW with the *sequential minimal optimization (SMO)* algorithm [14], using a third party *Matlab* SVM toolbox [3]. For the following experiments the SVM and kernel parameters were set to $C = 1$ and $\gamma = 1.8$, respectively.

In the first investigation we were concerned whether an SVM-GDTW is able to classify clearly separable data. Hence we applied the SVM-GDTW to character class pairs which (i) were shown to be *dissimilar* in respect to a certain measure [1] and (ii) achieved very low classification confusions with a traditional HMM based technique, developed by the authors and called statistical DTW (SDTW) [1]. Table 1 summarizes classification error rates and compares them to the results of SDTW. The table shows the satisfying result, that for the dissimilar character pairs (a ↔ b, d ↔ m) classification errors are rare in both classification methods, actually are due to mislabelings.

In a second category of two-class experiments the discrimination of character class pairs, which were shown to be *similar* and were frequently misclassified by SDTW, was examined. Table 1 illustrates, that for some of the selected character pairs (c ↔ e, b ↔ h) SVM-GDTW gives lower error rates than SDTW, for others (u ↔ v, y ↔ g) vice versa.

Table 1. Two-class experiments on UNIPEN data: error rate $E_{\text{SVM-GDTW}}$ of the SVM Gaussian DTW kernel approach for examples of *dissimilar* (a ↔ b, d ↔ m) and *similar* lower case character pairs (c ↔ e, u ↔ v, y ↔ g, b ↔ h). The number of training samples (M), support vectors (M_S) and the error rate E_{SDTW} for the generative, HMM-based approach of SDTW [1] are listed as well. Training set size was $66\,\%$, test set size $33\,\%$ of the UNIPEN Train-R01/V07 database.

Character pairs	M	M_S	$E_{\text{SVM-GDTW}}$	E_{SDTW} [1]
a ↔ b	3540	298	0.5 %	0.8 %
d ↔ m	2595	334	0.1 %	0.4 %
c ↔ e	5088	351	3.7 %	7.2 %
u ↔ v	2214	397	9.2 %	6.8 %
y ↔ g	2088	358	11.2 %	7.7 %
b ↔ h	2524	275	2.3 %	3.2 %

4.4. Multi-class experiments

For a multi-class experiment we have chosen the DAG-SVM approach [13]. DAG-SVM combines a set of two-class SVMs into a multi-class classifier. For a K-class problem DAG-SVM contains $K \cdot (K-1)/2$ two-class classifiers, one for each class pair. During classification $K - 1$ two-class SVM evaluations are combined using a decision directed acyclic graph (DDAG) topology.

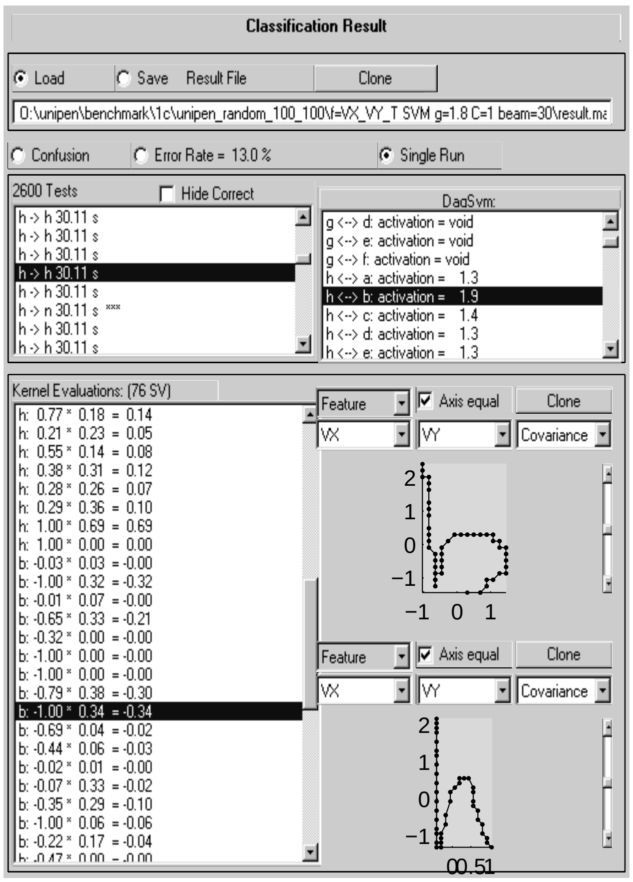
For the multi-class case we used smaller UNIPEN subsets due to efficiency reasons (see section 4.5). We made experiments on two different dataset sizes in order to give an idea of the recognizer's dependence on this quantity. Figure 2 gives a graphical illustration of an example classification showing a snapshot of our classification GUI.

Table 2 summarizes classification error rates of a DAG-SVM-GDTW classifier for the 1a/b/c UNIPEN sections and compares this result to other results on UNIPEN data collected from the literature. 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. Thus in the following comparison we refer only to the experiments on a unique set. In the table we typed these values boldface. From the values it can be seen that DAG-GDTW-SVM achieves lower error rates than SDTW for the relative small training set. For the larger training sets DAG-GDTW-SVM and SDTW achieve comparable error rates.

The higher performance of DAG-GDTW-SVM in comparison with SDTW on the small training set supports the statement that the statistical approach SDTW is highly dependent on accurate models, which cannot be satisfactorily gained from the relative small training set.

4.5. Complexity

A kernel evaluation (7) for a typical character pair asymptotically takes $C_{T_{\text{kernel}}} = \mathcal{O}(\tilde{N} \cdot F \cdot p)$ operations (with $\tilde{N} = 45$ the average length of the sequences, $F = 3$ the dimension of $\mathbf{t}_i$ and p the average number of path hypotheses in the beam search). Experimentally we mea-



```
O:\unipen\benchmark\1c\unipen_random_100_100\f=VX_VY_T SVM g=1.8 C=1 beam=30\result.ma
Error Rate =  13.0 %

2600 Tests
h -> h 30.11 s
h -> h 30.11 s
h -> h 30.11 s
h -> h 30.11 s
h -> h 30.11 s
h -> n 30.11 s  ***
h -> h 30.11 s
h -> h 30.11 s

g <--> d: activation = void
g <--> e: activation = void
g <--> f: activation = void
h <--> a: activation =   1.3
h <--> b: activation =   1.9
h <--> c: activation =   1.4
h <--> d: activation =   1.3
h <--> e: activation =   1.3

Kernel Evaluations: (76 SV)
h:  0.77 *  0.18  =  0.14
h:  0.21 *  0.23  =  0.05
h:  0.55 *  0.14  =  0.08
h:  0.38 *  0.31  =  0.12
h:  0.28 *  0.26  =  0.07
h:  0.29 *  0.36  =  0.10
h:  1.00 *  0.69  =  0.69
h:  1.00 *  0.00  =  0.00
b: -0.03 *  0.03  = -0.00
b: -1.00 *  0.32  = -0.32
b: -0.01 *  0.07  = -0.00
b: -0.65 *  0.33  = -0.21
b: -0.32 *  0.00  = -0.00
b: -1.00 *  0.00  = -0.00
b: -1.00 *  0.00  = -0.00
b: -0.79 *  0.38  = -0.30
b: -1.00 *  0.34  = -0.34
b: -0.69 *  0.04  = -0.02
b: -0.44 *  0.06  = -0.03
b: -0.02 *  0.01  = -0.00
b: -0.07 *  0.33  = -0.02
b: -0.35 *  0.29  = -0.10
b: -1.00 *  0.06  = -0.06
b: -0.22 *  0.17  = -0.04
b: -0.47 *  0.00  = -0.00
```

Figure 2. A snapshot from our multi-class DAG-SVM-GDTW classification GUI: it shows the classification of a test pattern $\mathcal{T}$ (of class "h"; selected in the upper left list). $K \cdot (K-1)/2$ two-class SVMs were trained with the DAG-SVM algorithm [13]. For illustration purposes the two-class SVM for the class pair (h $\leftrightarrow$ b) is selected in the upper right list. The activation $f(\mathcal{T})$ of this SVM is 1.9, hence $\mathcal{T}$ is correctly classified as the positive class "h". In the lower part of the figure the terms $S_i\alpha_i * K(\mathcal{T}, \mathcal{P}_i)$ for $i = 1, \ldots, M_S$ are listed on the left, each of which is the contribution of a support vector to the classification criterion (3). E.g., for the selected support vector (of class "b") $S_i = -1$, $\alpha_i = 1.00$ and $K(\mathcal{T}, \mathcal{P}_i) = 0.34$. To the right a graphical presentation of the selected support vector $\mathcal{P}_i$ and the test pattern $\mathcal{T}$ is illustrated.

Table 2. Multi-class experiments on various sections of UNIPEN data: error rate E of the DAG-SVM-GDTW approach and other classification techniques collected from the literature. In our experiments E is the mean of five experiments on different dataset combinations of equal size.

UNIPEN section	Approach	Error rate E	UNIPEN Database Type
1a (digits)	DAG-SVM-GDTW	**4.0 %** **3.8 %**	Train-R01/V07 rand. chosen 20 %/20 % Train/Test rand. chosen 40 %/40 % Train/Test
	SDTW [1]	**4.5 %** **3.2 %**	Train-R01/V07 rand. chosen 20 %/20 % Train/Test rand. chosen 40 %/40 % Train/Test
	MLP [12]	3.0 %	DevTest-R02/V02
	HMM [9]	3.2 %	Train-R01/V06 4 % "bad characters" removed
1b (upper case)	DAG-SVM-GDTW	**7.6 %** **7.6 %**	Train-R01/V07 rand. chosen 20 %/20 % Train/Test rand. chosen 40 %/40 % Train/Test
	SDTW [1]	**10.0 %** **8.0 %**	Train-R01/V07 rand. chosen 20 %/20 % Train/Test rand. chosen 40 %/40 % Train/Test
	HMM [9]	6.4 %	Train-R01/V06 4 % "bad characters" removed
1c (lower case)	DAG-SVM-GDTW	**11.7 %** **12.1 %**	Train-R01/V07 rand. chosen 10 %/10 % Train/Test rand. chosen 20 %/20 % Train/Test
	SDTW [1]	**13.0 %** **11.4 %** 9.7 %	Train-R01/V07 rand. chosen 10 %/10 % Train/Test rand. chosen 20 %/20 % Train/Test rand. chosen 66 %/33 % Train/Test
	MLP [12]	14.4 %	DevTest-R02/V02
	HMM-NN hybrid [6]	13,2 %	Train-R01/V07
	HMM [9]	14,1 %	Train-R01/V06 4 % "bad characters" removed

sured $C_{T_{\text{kernel}}} \approx 0.001\,\text{sec}$ in our implementation on an AMD Athlon 1200MHz.

The asymptotic training time of the two-class SMO and 26-class DAG-SVM training algorithm is $C_{T_{\text{train},2-\text{class}}} = \mathcal{O}(M^\gamma)$ and $C_{T_{\text{train},26-\text{class}}} = \mathcal{O}\left(2^{\gamma-1}K^{2-\gamma}M^\gamma\right)$, respectively, with $\gamma \approx 2$ and M the *total* number of training examples [13]. In the 1c multi-class experiments (training/test set size 20 % each) the complexities were measured as $C_{T_{\text{train},2-\text{class}}} \approx 0.25\,\text{h}$ and $C_{T_{\text{train},26-\text{class}}} \approx 81\,\text{h}$.

The average number of support vectors in the 1c multi-class experiments was $M_S \approx 100$. Classification time for one two-class SVM is $C_{T_{\text{test},2-\text{class}}} \approx M_S \cdot C_{T_{\text{kernel}}} = 100 \cdot 0.001\,\text{sec} = 0.1\,\text{sec}$. Since the DAG-SVM evaluates $K-1$ two-class SVMs, a classification of a lower case character takes $C_{T_{\text{test},26-\text{class}}} \approx (K-1) \cdot C_{T_{\text{test},2-\text{class}}} = 25 \cdot 0.1\,\text{sec} = 2.5\,\text{sec}$.

The memory complexity C_M mainly consists of the storage of all support vectors, thus $C_M = K \cdot (K-1)/2 \cdot M_S \cdot \tilde{N} \cdot F \cdot 4\,\text{byte} = 26 \cdot 25/2 \cdot 100 \cdot 45 \cdot 3 \cdot 4\,\text{byte} \approx 17.5\,\text{Mbyte}$.

Of course both time and memory complexity are immense and not practical for an operation of the DAG-SVM-GDTW classifier on handheld devices. However, we see some possibilities for optimization. A subsampling of the writing scales C_T and C_M linearly and—by a factor 3–5—might degrade recognition accuracy not too much. Additionally one can omit or interrupt applicable kernel evaluations by pruning techniques for the DTW. Furthermore techniques exist that decrease the number of support vectorsposterior to the SVM training [16]. These techniques produce SVMs which are up to ten times faster without large losses in classification accuracy.

5. Conclusion

We have presented a novel approach for the recognition of on-line handwritten characters. This technique combines dynamic time warping (DTW) and support vector machines (SVM) by integrating DTW into a Gaussian SVM kernel. The benefit of this approach is the absence of restrictive assumptions about class conditional densities, as made in conventional HMM based techniques. The only essential assumption made is the selection of the kernel.

We have applied the proposed classification technique to characters of the UNIPEN handwriting recognition database. Experiments have shown superior recognition rate in comparison to an HMM-based classifier for relative small training sets and comparable rates for larger training sets.

A problem of this approach is the complexity. However, we see possibilities to cope with this issue by several techniques, e.g. by a subsampling and by reducing the number of kernel evaluations and support vectors.

Further perspective and attractive challenges for future research are the establishment of a kernel based approach for word recognition and the analysis of other kernels, e.g. the Fisher kernel, set-kernels and other distance-based kernels.

It should be stated that the proposed approach can also be applied in other applications where the data is a variable-size sequence of feature vectors, like speech recognition and genome processing.

References

[1] C. Bahlmann and H. Burkhardt. Measuring HMM similarity with the Bayes probability of error and its application to online handwriting recognition. In *Proc. of the 6th ICDAR*, pages 406–411, 2001. 2.1, 4.3, 1, 2

[2] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998. 1, 2.2, 2.2

[3] G. C. Cawley. MATLAB support vector machine toolbox (v0.50β). University of East Anglia, School of Information Systems, Norwich, Norfolk, U.K. NR4 7TJ, 2000. URL http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox. 4.3

[4] N. Cristianini and J. Shawe-Taylor. *Support Vector Machines*. Cambridge University Press, 2000. 1, 2.2, 2.2

[5] D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1/3):161, 2002. 1, 3

[6] N. Gauthier, T. Artères, B. Dorizzi, and P. Gallinari. Strategies for combining on-line and off-line information in an on-line handwriting recognition system. In *Proc. of the 6th ICDAR*, pages 412–416, 2001. 2

[7] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proc. of the 12th ICPR*, pages 29–33, 1994. 1, 4.1

[8] B. Haasdonk and D. Keysers. Tangent distance kernels for support vector machines. In *Proc. of the 16th ICPR*, 2002. 3

[9] J. Hu, S. G. Lim, and M. K. Brown. Writer independent on-line handwriting recognition using an HMM approach. *Pattern Recognition*, 33:133–147, Jan. 2000. 2

[10] T. Jaakkola, M. Diekhans, and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. In T. Lengauer et al., editors, *Proc. 7th Int. Conf. on Intelligent Syst. for Molecular Biology (ISMB-99)*, 1999. 1

[11] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press/A Bradford Book, 1998. 2.1

[12] M. Parizeau, A. Lemieux, and C. Gagné. Character recognition experiments using UNIPEN data. In *Proc. of the 6th ICDAR*, pages 481–485, 2001. 2

[13] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGS for multiclass classification. In S. Solla, T. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*. MIT Press, 2000. 2.2, 4.4, 2, 4.5

[14] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*, chapter 12, pages 185–208. MIT Press, 1999. 4.3

[15] L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993. 2.1, 2.1

[16] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space versus feature space in kernel-based methods. *IEEE Trans. Neural Networks*, 10(5):1000–1017, Sept. 1999. 4.5

[17] C. Tappert. Cursive script recognition by elastic matching. *IBM J. Res. Develop.*, 26(6):765–771, Nov. 1982. 2.1

[18] V. Vuori, M. Aksela, J. Laaksonen, and E. Oja. Adaptive character recognizer for a hand-held device: Implementation and evaluation setup. In *Proc. of the 7th IWFHR*, pages 13–22, 2000. 2.1

[19] C. Watkins. Dynamic alignment kernels. In *Advances in Large Margin Classifiers*, pages 39–50. MIT Press, 2000. 1