Support Vector Machines and Dynamic Time Warping for Time Series

Steinn Gudmundsson Thomas Philip Runarsson Sven Sigurdsson

***Abstract*— Effective use of support vector machines (SVMs) in classification necessitates the appropriate choice of a kernel. Designing problem specific kernels involves the definition of a similarity measure, with the condition that kernels are positive semi-definite (PSD). An alternative approach which places no such restrictions on the similarity measure is to construct a set of inputs and let each example be represented by its similarity to all the examples in this set and then apply a conventional SVM to this transformed data. Dynamic time warping (DTW) is a well established distance measure for time series but has been of limited use in SVMs since it is not obvious how it can be used to derive a PSD kernel. The feasibility of the similarity based approach for DTW is investigated by applying the method to a large set of time-series classification problems.**

I. Introduction

THE support vector machine (SVM) has over the last decade become a popular approach to pattern classification since it can deliver state-of-the-art performance on a wide variety of real-world classification problems [1].

The kernel function is at heart of SVMs. This function is essentially a similarity measure between the objects under consideration. Kernels can be defined for general data types such as trees, strings and text, i.e. they are not limited to traditional vectorial data.

In the standard setting, the kernel is required to be symmetric and positive semi-definite (PSD). The latter requirement is rather strict, since many commonly used similarity and distance measures do not (easily) lead to PSD kernels. Construction of PSD kernels for a specific problem is unfortunately non-trivial.

A systematic way of including general similarity measures in SVMs is highly desirable since a wealth of such (dis)similarity measures already exists for a wide variety of objects.

For general (dis)similarity measures k-nearest neighbor (k-NN) algorithms are a natural choice. A specific example is 1-NN with dynamic time warping distance which has been found to work very well on many time series classification problems [2]. In general, $k-$NN classifiers work reasonably well but are known to be sensitive to noise in the training data such as irrelevant inputs and outliers. Since SVMs often outperform $k-$NNs on practical classification problems where a natural choice of PSD kernels exists (see [3] for an informative discussion on the subject) it is of considerable interest to extend the set of available kernels.

Several ad-hoc strategies have been proposed for including non-PSD "kernels" in SVMs. The simplest strategy is to simply ignore the fact that the kernel is non-PSD and see how it performs. In this case the existence of a Reproducing Kernel Hilbert Space is not guaranteed [4] and it is no longer clear what it is that is being optimized. Furthermore, the resulting optimization problem may no longer be convex, making it difficult to solve. Another strategy is to apply regularization to the kernel matrix, i.e. transform it in some way to make it PSD. This strategy is not very satisfying since it can lead to kernel matrices with large diagonal entries, resulting in overfitting [5]. In addition, it is not clear how to treat test examples which are not available at training time.

A simple strategy which can be applied to general pairwise similarity measures was proposed in [6]. It involves the construction of a set of inputs such that each example is represented with its similarity to all the examples in this set. An SVM is then applied to the transformed data in the usual way using this set as a training set. As a consequence, sparsity of the solution may be lost. Subsequent classification may thus become expensive since most of the training examples end up as support vectors. Some possibilities in retaining sparsity are described in the discussion section. The loss of sparsity is offset by the larger selection of available proximity measures.

Support vector machines trained on pairwise similarities have been employed with quite some success in computational biology (c.f. [7]) but the strategy appears not to be in widespread use in the time series community. The aim of this paper is to investigate the effectiveness of this strategy for time series classification when the pairwise similarities are based on the dynamic time warping distance.

The paper is organized as follows. In the next section the construction of the similarity based inputs is described in more detail and a connection made with the so-called arbitrary-kernel SVM. Section III details the dynamic time warping distance. Numerical experiments are presented in section IV and section V concludes the paper.

II. Support vector machines

First, a brief word about the notation employed in the paper. Matrices are represented as bold upper case characters ($\mathbf{A}$), column vectors as bold lower case ($\mathbf{x}$), unless explicitly transposed ($\mathbf{x}^T$). A vector of zeros and ones are denoted by $\mathbf{0}$ and $\mathbf{1}$ respectively.

Steinn Gudmundsson and Sven Sigurdsson are with the Department of Computer Science, University of Iceland.

Thomas Runarsson is with the Department of Engineering, University of Iceland (email: tpr@hi.is.)

The projected was supported by The Icelandic Center for Research (RANNIS)

Let $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$ denote a set with m labelled examples, where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. The input domain $\mathcal{X}$ is not restricted to be a subset of $\mathbb{R}^n$, but can be any set, e.g. set of strings or graphs. In the following it is assumed that $\mathcal{Y} = \{-1, 1\}$ unless otherwise noted. Let $\mathbf{Y}$ denote a $m \times m$ diagonal matrix with $\mathbf{Y}_{ii} = y_i$.

A *kernel* is a function $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ which for all $m \in \mathbb{N}$ and all $\mathbf{x}_1, \ldots, \mathbf{x}_m$ gives rise to a symmetric positive semi-definite matrix $\mathbf{K}$ with $\mathbf{K}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. The kernel implicitly defines a mapping from input space to *feature space*, $\mathbf{x} \mapsto \boldsymbol{\phi}(\mathbf{x})$. Two common kernel functions are the *linear kernel* $\kappa(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z}$ and the *Gaussian kernel* $\kappa(\mathbf{x}, \mathbf{z}) = \exp(-\gamma||\mathbf{x} - \mathbf{z}||^2)$ where $\gamma > 0$ is a user-specified shape parameter.

The 1-norm soft margin support vector machine is obtained by solving the following convex quadratic optimization problem [4]

$$\begin{array}{ll} \min_{\boldsymbol{\alpha}} & \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Y}\mathbf{K}\mathbf{Y}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{1} \\ \text{s.t.} & \boldsymbol{\alpha}^T\mathbf{Y}\mathbf{1} = 0 \\ & \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1} \end{array} \tag{1}$$

where C is a parameter which controls the trade off between maximizing the margin and allowing for incorrectly classified training examples and $\boldsymbol{\alpha} \in \mathbb{R}^m$ are Lagrange multipliers. The decision rule is given by

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{m} y_i\alpha_i\kappa(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{2}$$

The threshold b is

$$b = y_j - \sum_{i=1}^{m} y_i\alpha_i\kappa(\mathbf{x}_i, \mathbf{x}_j) \tag{3}$$

for any j with $\alpha_j > 0$ but is usually computed by averaging over all such j. The computation of the kernel function can be quite expensive, e.g. for the DTW distance described below the cost is quadratic in the length of the time series. From (2) it is obvious that sparsity of the solution is desirable since kernel evaluations corresponding to training examples with $\alpha_j = 0$ can be omitted.

A. Kernels from pairwise data

Following [6], it is assumed that instead of a proper kernel function, all that is available is a *proximity function* $P : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$. No restrictions are placed on the function P, not symmetry nor even continuity. Define the data dependent mapping Φ_m by

$$\Phi_m : \mathbf{x} \mapsto (P(\mathbf{x}, \mathbf{x}_1), P(\mathbf{x}, \mathbf{x}_2), \ldots, P(\mathbf{x}, \mathbf{x}_m))^T \tag{4}$$

where $\mathbf{x}_i,\ i = 1, \ldots, m$ are the examples in S and represent each training example $\mathbf{x}_i$ by $\tilde{\mathbf{x}}_i = \Phi_m(\mathbf{x}_i)$ i.e. an $m-$dimensional vector containing proximities to all the examples in S. Let $\mathbf{P}$ denote the $m \times m$ matrix with entries $P(\mathbf{x}_i, \mathbf{x}_j),\ i, j = 1, \ldots, m$. Using the linear kernel on this data representation, the resulting kernel matrix becomes $\mathbf{K} = \mathbf{P}\mathbf{P}^T$. In this case the decision rule (2) simplifies to

$$f(\mathbf{x}) = \text{sgn}\left(\boldsymbol{\alpha}^T\mathbf{Y}\mathbf{P}\Phi_m(\mathbf{x}) + b\right) \tag{5}$$

All elements of $\Phi_m(\mathbf{x})$ must be computed when classifying a point $\mathbf{x}$. The above SVM formulation will be referred to as the pairwise proximity function SVM (ppfSVM) in the following. Note that in this case the problem (1) is equivalent to the dual problem

$$\begin{array}{ll} \min_{\mathbf{w}, \boldsymbol{\xi}, b} & C\mathbf{1}^T\boldsymbol{\xi} + \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{s.t.} & \mathbf{Y}(\mathbf{P}\mathbf{w} + b\mathbf{1}) + \boldsymbol{\xi} \geq \mathbf{1} \\ & \boldsymbol{\xi} \geq \mathbf{0}, \end{array} \tag{6}$$

where $\mathbf{w} = \mathbf{P}^T\mathbf{Y}\boldsymbol{\alpha}$ and $\boldsymbol{\xi} \in \mathbb{R}^m$ are slack variables. Computationally, it is however usually more efficient to solve (1). Also note that the distance between two points $\boldsymbol{\phi}(\mathbf{x})$ and $\boldsymbol{\phi}(\mathbf{z})$ in feature space can be computed as

$$\begin{aligned} d^2(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{z})) &= ||\boldsymbol{\phi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{z})||^2 \\ &= \kappa(\mathbf{x}, \mathbf{x}) + \kappa(\mathbf{z}, \mathbf{z}) - 2\kappa(\mathbf{x}, \mathbf{z}) \end{aligned} \tag{7}$$

and can in turn be compared with the original (DTW) distance.

Remark The ppfSVM is related to the so-called arbitrary-kernel SVM, a special case of the generalized support vector machines introduced in [8]. Following [8] let $k(\mathbf{A}, \mathbf{B})$ with $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times \ell}$ denote a function that maps $\mathbb{R}^{m \times n} \times \mathbb{R}^{n \times \ell}$ into $\mathbb{R}^{m \times \ell}$. In particular, $k(\mathbf{x}^T, \mathbf{z})$ is a real number, and if $\mathbf{X}$ is an $m \times d$ matrix such that the i-th row contains the d-dimensional training example $\mathbf{x}_i$ then $k(\mathbf{x}^T, \mathbf{X}^T)$ is an m element row vector where element i is the function value between $\mathbf{x}$ and element i in the training set. $k(\mathbf{X}, \mathbf{X}^T)$ is thus an $m \times m$ matrix corresponding to the training set. The name *arbitrary-kernel* stems from the fact that no restrictions such as positive semi-definiteness, differentiability or continuity are put on the function k.

The training data is used to find a nonlinear separating surface of the form $k(\mathbf{x}^T, \mathbf{X}^T)\mathbf{Y}\mathbf{u} = \gamma$ which discriminates between the two classes -1 and 1. The decision rule is therefore

$$h(\mathbf{x}) = \text{sgn}\left(k(\mathbf{x}^T, \mathbf{X}^T)\mathbf{Y}\mathbf{u} - \gamma\right) \tag{8}$$

The parameters $\mathbf{u} \in \mathbb{R}^m$ and $\gamma \in \mathbb{R}$ are found by solving the following convex quadratic program ([8], a special case of eq. (8.6))

$$\begin{array}{ll} \min_{\mathbf{u}, \boldsymbol{\xi}, \gamma} & C\mathbf{1}^T\boldsymbol{\xi} + \frac{1}{2}\mathbf{u}^T\mathbf{u} \\ \text{s.t.} & \mathbf{Y}(k(\mathbf{X}, \mathbf{X}^T)\mathbf{Y}\mathbf{u} - \gamma\mathbf{1}) + \boldsymbol{\xi} \geq \mathbf{1} \\ & \boldsymbol{\xi} \geq 0, \end{array} \tag{9}$$

with $\mathbf{u} \in \mathbb{R}^m$, $\gamma \in \mathbb{R}$ and slack variables $\boldsymbol{\xi} \in \mathbb{R}^m$. Identifying $k(\mathbf{x}, \mathbf{z})$ with $p(\mathbf{x}, \mathbf{z})$, $k(\mathbf{x}, \mathbf{X}^T)$ with $\Phi_m^T(\mathbf{x})$, $k(\mathbf{X}, \mathbf{X}^T)$ with $\mathbf{P}$, $\mathbf{u}$ with $\mathbf{Y}\mathbf{w}$ and γ with $-b$ this problem is seen to be identical with (6).

III. Dynamic time warping

Let $\mathbf{p} = (p_1, p_2, \ldots, p_N)$ and $\mathbf{q} = (q_1, q_2, \ldots, q_N)$ denote two time series of length N. One of the simplest distance measures for time series is the Euclidian distance, $d_E(\mathbf{p}, \mathbf{q}) = ||\mathbf{p} - \mathbf{q}||_2$. The use of Euclidian distance for time series is problematic since it is sensitive to offset, amplitude

scaling, noise, phase shifts and temporal distortion. The first two concerns can be mitigated by normalization of individual time series but the other problems still remain.

Dynamic time warping (DTW) allows local contraction and expansion of the time axis, alleviating the alignment problem inherent with Euclidian distance. In the past, DTW was widely used in speech recognition and more recently in various time series data mining applications. It is a reasonable choice if prior knowledge about the data at hand is limited.

The first step in computing the DTW distance between time series $\mathbf{p}$ and $\mathbf{q}$, is to construct a $N \times N$ distance matrix $\mathbf{D}$ containing pairwise distances between the samples where $D_{ij} = (p_i - q_j)^2$ is commonly used, $i, j = 1, \ldots, N$. The objective is to find a path through the matrix so that the cumulative distance between $\mathbf{p}$ and $\mathbf{q}$ is minimized. This so-called warping path is constrained in order to make the optimization problem manageable and the resulting solution sensible. The warping path must start at $(1, 1)$ and end at (N, N), it should be continuous and movement backwards in time is prohibited. The amount of warping is sometimes further restricted by a *warping window* such as the Sakoe-Chiba band [9]. No warping window is used in the following. Let W_{ij} denote the DTW distance between subsequences $p_1, \ldots, p_i$ and $q_1, \ldots, q_j$. The DTW distance is obtained by solving the following recurrence relation

$$W_{ij} = D_{ij} + \min\left(W_{i-1,j}, W_{i-1,j-1}, W_{i,j-1}\right)$$

with $d_{DTW}(\mathbf{p}, \mathbf{q}) = \mathbf{W}_{NN}$.

Several attempts have been made to derive kernels based on the dynamic time warping distance. In [10] the *Gaussian dynamic time warping* (GDTW) kernel is defined as

$$k_{GDTW}(\mathbf{x}, \mathbf{z}) = \exp\big(-\gamma d_{DTW}(\mathbf{x}, \mathbf{z})\big).$$

A SVM with GDTW applied to a handwriting recognition task achieved superior recognition rates compared to a conventional hidden Markov model (HMM) based method. The authors point out that the kernel is not positive definite but offer some theoretical and empirical explanations to why it sometimes works well in practice.

In [11] the pairwise distances in the DTW algorithm are replaced with pairwise similarities and a dynamic programming algorithm used to find a path through the similarity matrix which makes the accumulated similarity as large as possible. The method is applied to a speaker-dependent speech recognition problem and is found to have comparable recognition performance to HMMs.

A different approach is taken in [12]. The authors consider the similarity score spanned by all possible series alignments, instead of a single highest score. The authors provide a proof that their kernel is positive definite and point out that the kernel in [11] is not PSD. Although the kernel in [12] is PSD, the resulting kernel matrix has very large entries on the diagonal (compared to the off-diagonal entries) and the resulting classifier may have poor generalization properties [5]. The remedy employed in [12] is to use the logarithm of the kernel values instead. Since this may result in a matrix that is no longer positive definite, the training matrix is regularized by subtracting the smallest eigenvalue from the diagonal while the test-train matrix is left unchanged.

Arguably, the simplest way to define a "kernel" based on d_{DTW} is

$$k_{NDTW}(\mathbf{x}, \mathbf{z}) = -d_{DTW}(\mathbf{x}, \mathbf{z})$$

which will be referred to as the *negated dynamic time warping* (NDTW) kernel in the following, and can be shown to be indefinite via a simple example. The other kernel considered here is the Gaussian DTW kernel from [10]. When reference is made to GDTW and NDTW as "kernels" below, they are understood to be possibly non-PSD.

IV. Experiments

All experiments were carried out in Matlab using the LIBSVM package [13].

A. Gaussian kernel

The first task was to investigate the loss of accuracy caused by the pairwise proximity formulation when the proximity function P happens to be a positive semi-definite kernel, namely $P(\mathbf{x}, \mathbf{z}) = \kappa(\mathbf{x}, \mathbf{z}) = \exp(-\gamma||\mathbf{x} - \mathbf{z}||^2)$. In this case it is possible to compare the performance with a conventional SVM. A subset of the USPS dataset [14] containing digits 2 and 7 was used in the experiment. The training set had 1376 elements and the test set 345 elements.

Since SVMs are sensitive to the choice of parameters (C and γ in this case), model selection was performed for both methods. Five fold stratified cross-validation on the training set was used to search for the parameter values. The search was carried out for $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^1, 2^3\}$ and $C \in \{2^{-5}, 2^{-3}, \ldots, 2^{13}, 2^{15}\}$. After training a classifier using the "optimal" (C, γ) pair, its performance was evaluated on the test set.

Once the kernel matrices had been formed, the training and testing of the ppfSVM was slightly slower than for the conventional SVM (approx 13%.)

The standard SVM made 4 errors on the test set and the number of support vectors was 71. The ppfSVM made 5 errors indicating that loss in accuracy is acceptable. The number of support vectors was 41.

B. DTW distance

The UCR time series benchmark [15] was used to compare the performance of ppfSVM with SVM using the (indefinite) dynamic time warping kernels described above. The benchmark represents a wide variety of practical classification problems, including biomedical data, electromagnetic measurements of lightning activity, movement tracking in video surveillance and so on. The length of the time series varies from 60 to 637 so that computation of DTW is easily feasible. Each of the 20 data sets comes with a predefined training/test set partition. A description of the datasets is given in table I. The 50words dataset is omitted since it is not compatible with the one-against-one strategy for multi-class problems. For state of the art performance on this benchmark see [16].

Dataset	n_c	m	m_{te}	N
SynthControl	6	300	300	60
Gun Point	2	50	150	150
CBF	3	30	900	128
Face(all)	14	560	1690	131
OSULeaf	6	200	242	427
SwedishLeaf	15	500	625	128
Trace	4	100	100	275
TwoPatterns	4	1000	4000	128
Wafer	2	1000	6174	152
Face(four)	4	24	88	350
Lightning2	2	60	61	637
Lightning7	7	70	73	319
ECG200	2	100	100	96
Adiac	37	390	391	176
Yoga	2	300	3000	426
Fish	7	175	175	463
Beef	5	30	30	470
Coffee	2	28	28	286
OliveOil	4	30	30	570

TABLE I

STATISTICS FOR THE UCR TIME SERIES DATA SETS. NUMBER OF CLASSES (n_c), SIZE OF TRAINING SET (m), SIZE OF TEST SET (m_{te}) AND LENGTH OF TIME SERIES (N).

Model selection is done by performing five fold stratified cross-validation on the training set, searching over the same range of parameter values as before (for NDTW the value of γ does not matter.) After training a classifier using the best parameter values, its performance was evaluated on the independent test set. The results of SVM and ppfSVM on the UCR benchmark are given in table II. A nearest neighbor classifier with the DTW distance is included as a baseline. The classifiers were ranked on each dataset according to their performance and the ranks averaged over all datasets (lower rank indicates better performance.)

The performance of the nearest neighbor classifier and ppfSVM/NDTW is practically identical (judged by the rank averages), with the ppfSVM/GDTW trailing slightly behind. The nonparametric Bonferroni-Dunn test [17] was used to compare the nearest neighbor classifier with the others. At significance level $\alpha = 0.05$, the critical value of the test is 1.28. Only SVM/NDTW performs significantly worse than NN but SVM/GDTW is just below significant difference.

The ppfSVM classifier does not appear to break down on any of the data sets, except on Lightning2, but this is probably an artifact of the small test set. In contrast, plugging the two kernels directly in a SVM results in occasional breakdowns. For NDTW this is most evident with the Gun-Point, OSULeaf, SwedishLeaf, ECG200, Adiac and Yoga datasets. The SVM with GDTW suffers on SwedishLeaf and Adiac. The last two problems are difficult since they have a large number of classes compared to the number of training examples.

The inferior performance of SVM with NDTW and GDTW does not come as a surprise since the kernel matrix was almost always indefinite. Further investigations reveal that regularization of indefinite kernel matrices, where the smallest eigenvalue is subtracted from the diagonal, does not help. The resulting classifiers actually had considerably lower overall accuracy than their non-regularized counterparts.

By smoothing out large distances, the GDTW kernel reduces the contribution of distant training examples on the decision boundary. This should increase robustness towards outliers compared to NDTW. Since ppfSVM/GDTW performs slightly worse than ppfSVM/NDTW it may suggest that outliers are not a problem. Another explanation is that the model selection may be failing to some extent, i.e. the amount of training data does not justify the tuning of two parameters with cross-validation. This is supported by the observation that optimizing the number of neighbors in a k-nearest neighbor classifier (via cross-validation) lead to a significantly higher error rate than for a simple nearest neighbor classifier (k fixed as 1.)

A further complication arises in the case of the SVMs. In some cases the number of available training examples for a single class is so small that the binary classification problems arising in the one-against-one procedure are heavily unbalanced, reducing accuracy of the final classifier.

Dataset	NN DTW	ppfSVM NDTW	ppfSVM GDTW	SVM NDTW	SVM GDTW
SynthControl	0.67	1.33	1.33	1.33	2.33
Gun Point	9.33	4.67	14.00	46.00	12.67
CBF	0.33	0.33	0.11	1.00	4.56
Face(all)	19.23	23.73	22.60	17.04	26.57
OSULeaf	40.91	40.50	35.54	70.66	40.08
SwedishLeaf	20.80	14.72	15.52	36.32	38.24
Trace	0.00	0.00	0.00	0.00	0.00
TwoPatterns	0.00	0.00	0.08	0.67	0.00
Wafer	2.01	1.05	1.52	18.12	3.39
Face(four)	17.05	14.77	11.36	10.23	11.36
Lightning2	13.11	32.79	31.15	49.18	11.48
Lightning7	27.40	30.14	31.51	21.92	30.14
ECG200	23.00	16.00	22.00	44.00	17.00
Adiac	39.64	32.48	34.27	51.15	56.01
Yoga	16.37	22.67	17.67	53.43	21.87
Fish	16.57	18.86	24.00	24.00	29.71
Beef	50.00	56.67	53.33	63.33	60.00
Coffee	17.86	10.71	17.86	50.00	17.86
OliveOil	13.33	16.67	26.67	13.33	26.67
Average rank	2.42	2.47	2.82	3.74	3.55

TABLE II

TEST ERROR (%) FOR THE UCR TIME SERIES DATA SETS.

Intuitively, if distances in feature space have little or no correlation with the original DTW distances, building a classifier on the approach described in this paper is not a promising task. In [18] the properties of various strategies for embedding DTW distances in feature space are investigated. The classification accuracy is conjectured to depend on the ability of the embedding method to maintain small distances while putting smaller weight on long distances. Figure 1 compares the original distances with distances in feature space based on equation (7). From this figure it appears that the relationship between classifier accuracy and distance is quite complex. Moreover, it turns out that the results are quite sensitive towards the choice of parameters, C and γ.

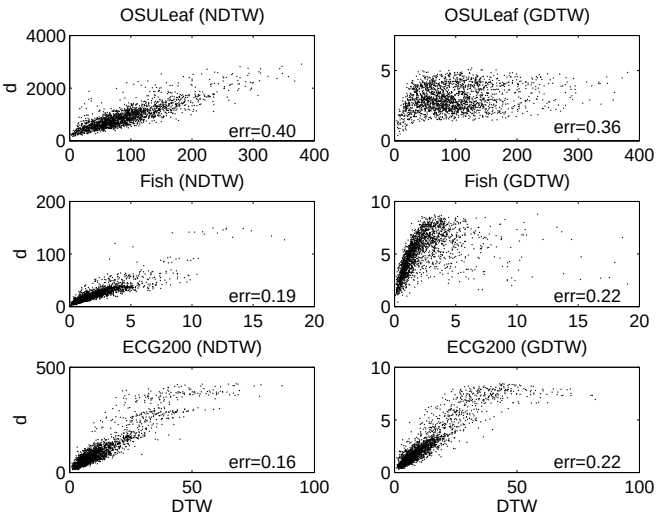

Fig. 1. Distances in feature space versus original DTW distances for the OSULeaf (top), Fish (middle) and ECG200 (bottom) data sets.

V. Discussion

The experiments with the UCR time series benchmark suggest that ppfSVM is competitive to the nearest neighbor classifier with dynamic time warping. A conventional SVM with the Gaussian DTW kernel performs significantly worse which illustrates that potential pitfalls of using indefinite kernels. The price to be paid is lack of sparsity in the decision rule with respect to kernel evaluations. The sparsity issue is addressed in [19] where a related method is given which enforces sparsity by modifying the optimization criteria and solving a linear programming problem. Another possibility would be to utilize the reduced support vector machine from [20].

The performance on the USPS data with a positive-definite kernel indicates that the ppfSVM does not suffer large loss in accuracy compared to a conventional SVM.

Directions of future work include experimenting with different embedding strategies such as [18] to improve accuracy. The unconstrained DTW algorithm is quadratic in the length of the time series. Coarse-graining or downsampling can be applied to long time series to make the DTW compations feasible. An alternative approach would be to construct a similarity measure based on lower bounds on the DTW distance such as [21] which can be computed in linear time.

Acknowledgments

The project was supported by The Icelandic Center for Research (RANNIS).

References

[1] I. Guyon, "SVM application list," http://www.clopinet.com/isabelle/Projects/SVM/applist.html.

[2] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*. New York, NY, USA: ACM, 2006, pp. 1033–1040.

[3] P. Vincent and Y. Bengio, "K-local hyperplane and convex distance nearest neighbor algorithms," in *Advances in Neural Information Processing Systems*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14. Cambridge, MA, USA: MIT Press, 2002, pp. 985–992.

[4] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[5] J. Weston, B. Schölkopf, E. Eskin, C. Leslie, and W. Noble, "Dealing with large diagonals in kernel matrices," *Annals of the Institute of Statistical Mathematics*, vol. 55, no. 2, pp. 391–408, 2003.

[6] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer, "Classification on pairwise proximity data," in *Advances in Neural Information Processing Systems*, vol. 11. Cambridge, MA, USA: MIT Press, 1999, pp. 438–444.

[7] L. Liao and W. S. Noble, "Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships," *Journal of Computational Biology*, vol. 10, no. 6, pp. 857–868, 2003.

[8] O. Mangasarian, "Generalized support vector machines," in *Advances in Large Margin Classifiers*, A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA, USA: MIT Press, 2000, pp. 135–146.

[9] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, speech, and signal processing*, vol. ASSP-26, no. 1, 1978.

[10] C. Bahlmann, B. Haasdonk, and H. Burkhardt, "On-line handwriting recognition with support vector machines—a kernel approach," in *Proc. 8th Int. Workshop Front. Handwriting Recognition (IWFHR)*, 2002, pp. 49–54.

[11] H. Shimodaira, K.-I. Noma, M. Nakai, and S. Sagayama, "Dynamic time-alignment kernel in support vector machine," in *Advances in Neural Information Processing Systems*, vol. 14. Cambridge, MA, USA: MIT Press, 2002, pp. 921–928.

[12] M. Cuturi, J.-P. Vert, O. Birkenes, and T. Matsui, "A kernel for time series based on global alignments," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, 2007, pp. 413–416.

[13] C. Chang and C. Lin, "LIBSVM: a library for support vector machines," Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[15] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "The UCR time series classification/clustering homepage," http://www.cs.ucr.edu/~eamonn/time_series_data/, 2006.

[16] J. Rodriguez and L. Kuncheva, "Time series classification: Decision forests and SVM on interval and DTW features," in *Proc Workshop on Time Series Classification, 13th International Conference on Knowledge Discovery and Data mining*, 2007.

[17] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *JMLR*, vol. 7, pp. 1–30, 2006.

[18] A. Hayashi, Y. Mizuhara, and N. Suematsu, "Embedding time series data for classification," in *Machine Learning and Data Mining in Pattern Recognition*, 2005, pp. 356–365.

[19] T. Graepel, R. Herbrich, B. Schölkopf, A. Smola, P. Bartlett, K.-R. Müller, K. Obermayer, and R. Williamson, "Classification on proximity data with LP-machines," in *Ninth International Conference on Artificial Neural Networks*. London: IEE, 1999, pp. 304–309.

[20] Y.-J. Lee and O. Mangasarian, "RSVM: Reduced support vector machines," in *Proceedings of the SIAM International Conference on Data Mining*. Philadelphia: SIAM, 2001.

[21] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and Information Systems*, vol. 7, no. 3, pp. 358–386, 2005.