Jalalian, Arash; Chalup, Stephan K. 'GDTW-P-SVMs: variable-length time series analysis using support vector machines', Neurocomputing Vol. 99, Issue 1, p. 270-282 (2013)

Available from: http://dx.doi.org/10.1016/j.neucom.2012.07.006

GDTW-P-SVMs: Variable-Length Time Series Analysis Using Support Vector Machines

Arash Jalalian* and Stephan K. Chalup

arash.jalalian@uon.edu.au, stephan.chalup@newcastle.edu.au
The University of Newcastle, School of Electrical Engineering and Computer Science
Callaghan, NSW, 2308, Australia. Tel: +61-249218628
** indicates the corresponding author*

Abstract

We describe a new technique for sequential data analysis, called GDTW-P-SVMs. It is a maximum margin method for the construction of classifiers with variable-length input series. It employs potential support vector machines (P-SVMs) and Gaussian dynamic time warping (GDTW) to waive the fixed-length restriction of feature vectors in training and test data. As a result, GDTW-P-SVMs enjoy the P-SVM method's properties such as the ability to: i) handle data and kernel matrices that are neither positive definite nor square and ii) minimise a scale-invariant capacity measure. The new technique elaborates on the P-SVM kernel functions, by utilising the well-known dynamic time warping algorithm to provide an elastic distance measure for the kernel functions. Benchmarks for classification are performed with several real-world data sets from the UCR Time Series Classification/Clustering page, the GeoLife trajectory data set, and the UCI Machine Learning Repository. The data sets include data with both variable and fixed-length input series. The results show that the new method performs significantly better than the benchmarked standard classification methods.

Keywords:
Support Vector Machines, Dynamic Time Warping, Time Series Classification, Sequential Data Analysis

---

1. Introduction

Within the context of time series analysis, sequential data classification has received great interest during the last decade. It has been widely applied to various research areas such as financial data mining [1, 2, 3], moving object identification [4, 5], medical data analysis [6, 7, 8], trajectory data analysis [9, 10, 11], time-stamped event data processing [12], and network monitoring [13, 14]. In all applications of sequential data classification using a kernel-based learning approach the data are represented in a new space by a similarity/distance measure. In the new space the data are aligned such that similar features correspond to each other. The features which represent the same property of the data are called *matching features/time stamps*. Different representations of the same sequential data could lead to different matching feature sets. This makes *sequential pattern matching* which includes comparing sequences of features for the presence of some pattern, a challenging problem.

Many distance measures have been proposed to solve the above-mentioned problem [15, 16, 17, 18]. In sequential pattern matching two types of distance measures have been employed: *elastic* [19] and *metric* [20] distance measures. According to these two types, common distance measures fall into three categories [18]:

1. Non-elastic metric (Euclidean Distance, l_p-norms and Correlation [19])
2. Elastic non-metric (Dynamic Time Warping [21] and Longest Common Sub-sequence [22])
3. Elastic metric (Edit Distance with Real Penalty [23])

Metric distance measures satisfy the *triangle inequality*[1]. This condition makes possible the efficient pruning of large numbers of time series that deviate too far from a matching pattern [24, 18]. Comprehensive applications have shown that Dynamic Time Warping (DTW) [25, 21, 26] among elastic non-metric distances and the Euclidean Distance among non-elastic metric distances outperformed most of the other distance measures [27, 28, 29, 30, 24].

DTW is a dynamic programming algorithm for measuring the distance between two sets of sequential data, which can be turned into a linear representation in time space (Figure 1). DTW has initially been proposed and used in automatic speech recognition [21]. It aims to align two sequences of input series by warping the time axis iteratively until an optimal match between the two sequences is found.

---

[1] $d(x, y) \leq d(x, z) + d(z, y)$ (*triangle inequality*).

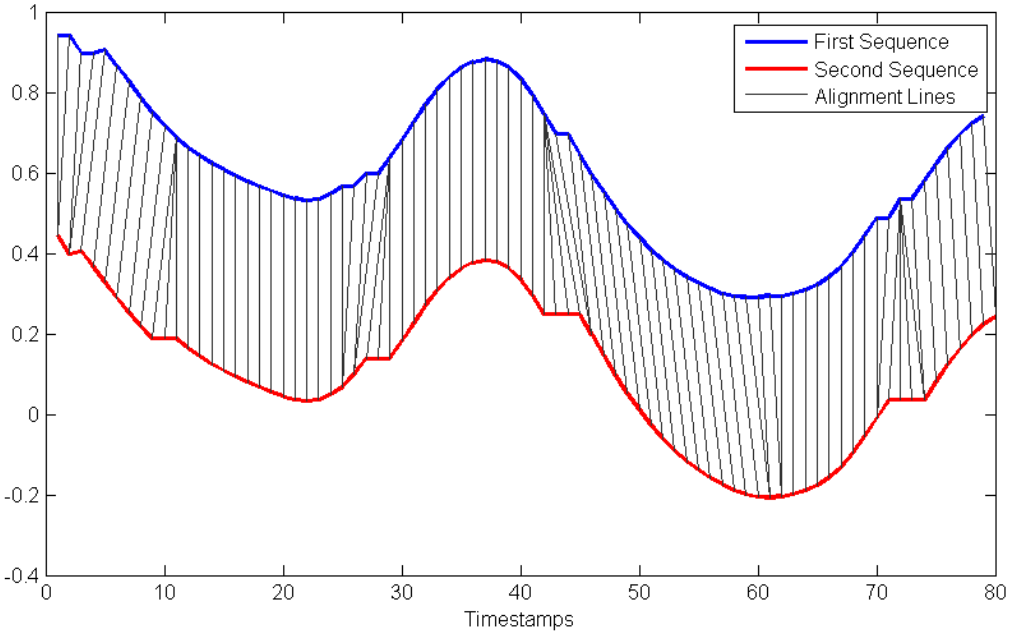

Figure 1: Sequence alignment using Dynamic Time Warping [21]; Illustration of comparing points in two sequences using Dynamic Time Warping. As shown, the two sequences have two different lengths. Finding the correspondence between data points has made DTW capable of comparing data objects with different lengths.

DTW is capable of an elastic and robust sequential data matching, and it tolerates variable sequence length which is common in sequential patterns (for example movement trajectories). Sequences are *warped/stretched* non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. DTW has been widely used as a distance measure for time series classification and clustering. A variety of the DTW algorithms have been proposed for different applications such as weighted dynamic time warping [24], derivative DTW [25], multidimensional DTW [31], DTW for pitch determination [32], scaling up DTW [33] and optimised DTW [34]. However, DTW does not account for the relative importance regarding the phase difference between a reference point and a testing point [24]. This may lead to misclassification, especially in applications where the shape similarity between two sequences is a major consideration for accurate recognition [24]. All of the above mentioned algorithms have employed DTW without using a learning algorithm. As we will show in our experiments, combining DTW with a learning algorithm helps to perform the classification of the sequential data with higher accuracy and overcomes DTW problems.

Support vector machines (SVMs), on the other hand, have become a popular approach to pattern classification since they can deliver state-of-the-art performance on a wide variety of real-world classification problems [35, 36, 37, 38]. Many interesting kernels have been proposed for sequential data classifications using SVMs [8, 39, 40, 20]. Mutual information kernels have been proposed for a special case of sequential string classification [8, 41]. These kernels are able to solve classification problems in high dimensional space where labelled data are sparse and unlabelled data are abundant [41]. Context-free models or probabilistic suffix tree structure have been employed to construct these kernels for an application of protein classification [8]. For solving general classification problems for variable length data objects, it is very tempting to plug the sequential distance measures into SVM kernels such as the Gaussian kernel [20].

In the present article advantages and disadvantages of several classification methods for variable-length input series are discussed and a new method is proposed to avoid the shortcomings of others. The article is structured as follows: Section 2 provides background on the existing classification methods. Section 3 introduces the new method called GDTW-P-SVM. The experiments are described in section 4 which is followed by a discussion and conclusion.

2. Background

2.1. Support Vector Machines

Suppose the space X, containing the input data, is referred to as the *input space*, while $F = \{\phi(\mathbf{x}) : \mathbf{x} \in X\}$ is an inner product space (Hilbert Space) and is called the *feature space*, where $\phi : X \longrightarrow F$ is the feature map from X to F. A *kernel* function [42] is a function $K : X \times X \longrightarrow \mathbb{R}$, such that for all $\mathbf{x}, \mathbf{z} \in X$

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \tag{1}$$

One of the most common kernel functions is the *Gaussian kernel* which is defined as:

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \, ||\mathbf{x} - \mathbf{z}||^2\right) \tag{2}$$

where $\gamma > 0$ is a user-specified shape parameter. Consider classifying a training sample $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l))$, using the feature space implicitly defined by the kernel $K(\mathbf{x}, \mathbf{y})$ and suppose the parameters α^* solve the following quadratic optimisation problem:

$$\begin{array}{ll} \textit{maximise} & W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j), \\ \textit{subject to} & \qquad \sum_{i=1}^{l} y_i \alpha_i = 0, \\ & \qquad C \geq \alpha_i \geq 0, i = 1, \ldots, l, \end{array} \tag{3}$$

where l is the number of training samples, y_i is the label for ith training sample and C is a real parameter, which is varied through a wide range of values while the optimal performance assessed using a separate validation set by cross-validation [42]. Let $f(\mathbf{x}) = \sum_{i=1}^{l} y_i \alpha_i^* K(\mathbf{x}_i, \mathbf{x}) + b^*$, where b^* is chosen so that $y_i f(\mathbf{x}_i) = 1$ for any i with $C > \alpha_i^* > 0$. Then the decision rule given by $sgn(f(\mathbf{x}))$ is equivalent to the hyperplane in feature space implicitly defined by the kernel $K(\mathbf{x}, \mathbf{z})$ that solves the optimisation problem (Equation 3), where b^* is chosen using the Karush-Kuhn-Tucker conditions [43].

Let $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a finite input space with $K(\mathbf{x}_i, \mathbf{x}_j)$ a symmetric function on X. Then $K(\mathbf{x}_i, \mathbf{x}_j)$ is a *Mercer kernel* if the matrix

$$\mathbf{K} = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^{n} \tag{4}$$

is positive semi-definite (PSD)[2] [45, 46]. Three approaches for data classification using SVMs have been proposed:

---

[2] A Hermitian matrix is PSD if all its eigenvalues are non-negative [44, 42].

1. If the data matrix (Equation 4) is PSD, it is interpreted as a *Gram matrix* and SVMs are subsequently applied [47].
2. If the data matrix is indefinite but symmetric, the matrix is projected into a subspace spanned by the eigenvectors with positive eigenvalues [48].
3. Another approach for dealing with indefinite data matrices involves flipping the sign of negative eigenvalues [49].

All three approaches guarantee a PSD matrix on the available training set but it may not be PSD on the new test set.

2.2. SVMs with GDTW kernel

The Gaussian function with Euclidean distance measure (Equation 2) is among the most commonly used kernels in SVMs. It is well-known that the Gaussian function provides a Mercer kernel [50]. It maps n vectors $v_1, v_2, \ldots, v_n$ into a Hilbert space [20] where $\phi(v_1), \phi(v_2), \ldots, \phi(v_n)$ span an n-dimensional subspace [50]. The Euclidean distance measure used in Equation 2 is able to compare two vectors with the same length only. Therefore, the classifier that is using this kernel function is restricted to an input space with fixed-length feature vectors. For instance, sequential data that vary in speed or time (such as: pedestrian's trajectory data and human voice data with variable recording times) cannot be directly used as the input space for this kernel function.

In order to overcome this problem, a Gaussian function can be defined with a DTW distance measure [51, 52]:

$$k_{GDTW}(\mathbf{x}^r, \mathbf{y}^s) = \exp\left(-\frac{D(\mathbf{x}^r, \mathbf{y}^s)}{\sigma^2}\right) \tag{5}$$

where $\mathbf{x}^r$ is a time series with discrete time index varying between 1 and r, $\mathbf{y}^s$ is a time series with discrete time index varying between 1 and s, σ is the Gaussian kernel width, and $D(\mathbf{x}^r, \mathbf{y}^s)$ is the DTW distance. It can be calculated recursively as [18]:

$$D(\mathbf{x}^r, \mathbf{y}^s) = ||x_r - y_s||_p + min \begin{cases} D(\mathbf{x}^{r-1}, \mathbf{y}^s) & delete, \\ D(\mathbf{x}^{r-1}, \mathbf{y}^{s-1}) & match, \\ D(\mathbf{x}^r, \mathbf{y}^{s-1}) & insert, \end{cases} \tag{6}$$

where $x_r \in \mathbb{R}^d$ is the rth element (last element) of time series $\mathbf{x}^r$, $y_s \in \mathbb{R}^d$ is the sth element (last element) of time series $\mathbf{y}^s$, and $||x_r - y_s||_p$ is the l_p-norm

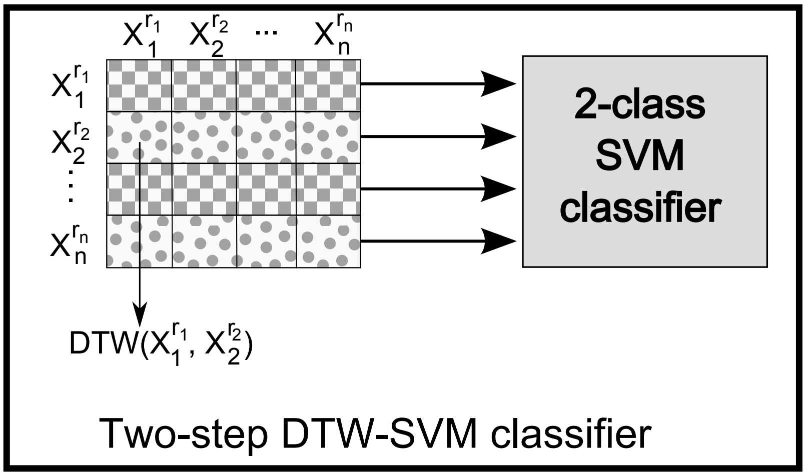
$X_1^{r_1}$ $X_2^{r_2}$ ... $X_n^{r_n}$
$X_1^{r_1}$
$X_2^{r_2}$
$X_n^{r_n}$
2-class SVM classifier
DTW($X_1^{r_1}$, $X_2^{r_2}$)
Two-step DTW-SVM classifier

Figure 2: Two-step DTW-SVMs methodology for classifying two classes of data with variable-length input series. It is used along with the pairwise classification method for multi-class classification problems. $\mathbf{x}_1^{r_1}, \mathbf{x}_2^{r_2}, \ldots, \mathbf{x}_n^{r_n}$ are data samples with different lengths, and $DTW(\mathbf{x}_1^{r_1}, \mathbf{x}_2^{r_2})$ is the DTW distance between $\mathbf{x}_1^{r_1}$ and $\mathbf{x}_2^{r_2}$.

in $\mathbb{R}^d$. For further information about DTW algorithms and their variations please refer to [18, 21, 53].

2.3. Two-step DTW-SVMs

The effective use of SVMs in classification necessitates the appropriate choice of a kernel. Classifying data sets that contain variable-length input series requires designing problem specific kernels. This involves the definition of a similarity measure, with the condition that the kernels are PSD. An alternative technique is discussed here, which uses a two-step architecture for classifying the data.

In the first step of this classification technique, the data has been represented by the DTW distance measure. DTW is able to find the distance between two input series with different lengths. Each sample is represented by its DTW distances to all other data samples. This is shown as a matrix in Figure 2 and we call it the *DTW matrix*. In this matrix each row/column is representing a transformed data sample. Since $DTW(\mathbf{x}_i^{r_i}, \mathbf{x}_j^{r_j}) = DTW(\mathbf{x}_j^{r_j}, \mathbf{x}_i^{r_i})$, the matrix is symmetric.

In the second step the DTW matrix is used as the input of a standard two-class SVM classifier (as shown in Figure 2). This technique can be used

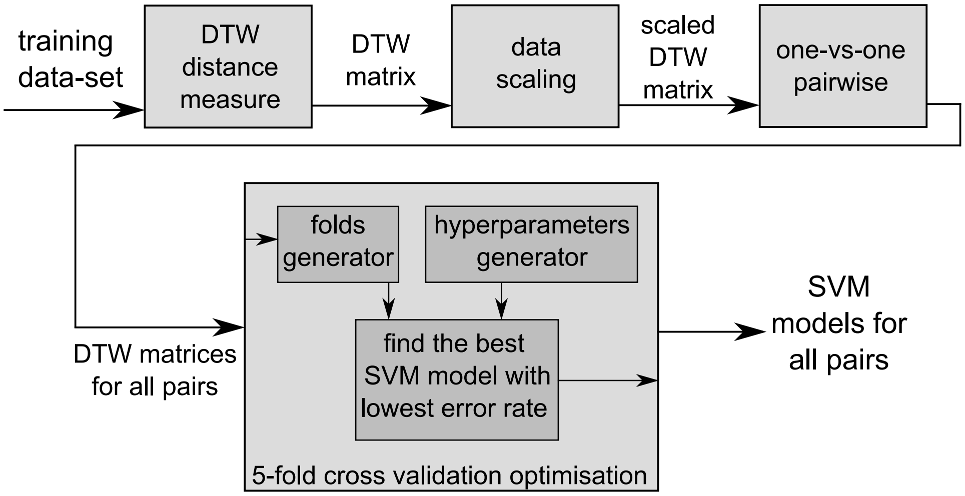

Figure 3: Training phase for multi-class classification problems using the idea presented in Figure 2; the pairwise classification algorithm (one-vs-one) is employed to train multi-class data, 5-fold cross validation with a leave-one-out policy is utilised for tuning hyperparameters (C, γ and kernel parameters).

along with the pairwise classification algorithm to classify multi-class time series.

Figure 3 shows different stages in the *training phase* of the multi-class classification method using the two-step DTW-SVMs classifier and a pairwise one-vs-one algorithm. A 5-fold cross validation technique is also employed to *tune* the SVM hyperparameters (C, γ and kernel parameters). As shown, after calculating the DTW distances between all samples the distances are scaled in $[0, 1]$ (The scaling method is described in Section 4, Equation 10). Then pairs are created using the scaled DTW matrix and class labels. Each pair contains two classes of data only (one-vs-one pairwise algorithm). Afterwards, 5-fold cross validation is utilised to tune the hyperparameters. As a result of this tuning, one SVM model for each pair of classes will be constructed.

The *testing phase* in this classification technique, as shown in Figure 4, is different from the testing phase in commonly used classification methods. In the training phase we used all training data to obtain the DTW matrix and represent the input space for the SVM classifier. In the test phase each testing object has to be mapped with the same representation as it was used in the

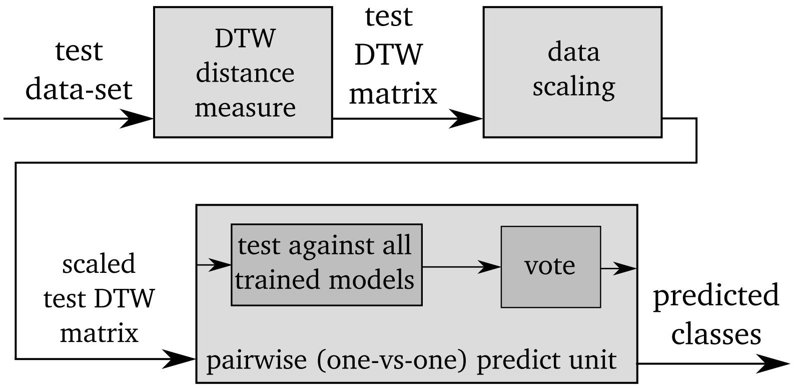

Figure 4: Testing phase for trained models obtained using the architecture shown in Figure 3; a pairwise classification algorithm (one-vs-one) is employed to label multi-class data.

training phase. This means to classify a test sample, the distances between the test sample and all samples in the training set have to be calculated. The matrix that contains these distances is called the *test DTW matrix* in Figure 4. The i^{th} row of the matrix represents the DTW distances between the i^{th} test sample and all samples in the training set.

This technique overcomes the problem of classifying data samples with different lengths using SVMs, and it enjoys the benefits of using the DTW distance measure without suffering from employing non-PSD kernels in SVMs. However in the testing phase, to calculate the test DTW matrix (as shown in Figure 4) the training set as well as the trained models are required for classification. This makes it difficult to distribute trained data. Either because they are too big to distribute or in some cases the training data sets are not allowed to be accessed by a testing party. Also, the DTW distances between each test sample and all training samples have to be calculated, which obviously slows down the testing process. Therefore this method is not technically feasible for applications that have a big training set or require real-time classification.

3. Proposal of GDTW-P-SVMs

3.1. Positive semi-definiteness and the GDTW kernel

As noted by [51] and supported by the proof provided by [54], k_{GDTW} cannot be a PSD function in general. Positive definiteness of k_{GDTW} depends on the DTW distances between the data objects and we cannot say that the GDTW function always satisfies or always does not satisfy Mercer's condition. The trainability of SVMs with the GDTW kernel is compared with the proposed method in Section 4.1.1.

Using the GDTW kernel in SVMs may result in a non-PSD kernel and in this case the existence of a Reproducing Kernel Hilbert Space is not guaranteed [55]. Another approach is to apply a transformation to the kernel matrix and make it PSD. This approach can lead to kernel matrices with large diagonal entries, resulting in overfitting [56]. Also it is not clear how this approach can handle new data objects in the test set [57]. As discussed in [57], fixing the diagonal values by subtracting the smallest eigenvalue from the diagonal does not increase the accuracy of the resulting classifier.

3.2. Combination of GDTW and P-SVMs

Employing DTW distance as a distance measure in Gaussian Kernel Functions and obtaining a new kernel function (Equation 5) called GDTW, is a tempting solution to waive SVM restriction on length of feature vectors. As discussed (above section), it is not clear under what conditions the GDTW function (Equation 5) satisfies Mercer's conditions and could be considered as a valid kernel function for SVMs.

On the other hand, the discussed two-step DTW-SVMs classification method has two main shortcomings: i) It needs the training data sets as well as the trained model when testing a new sample, and ii) while online testing is a common requirement of many time series classification problems such as speech recognition and handwriting recognition, testing against a large training data set using the two-step DTW-SVMs classifier (as shown in Figure 4) could be a very slow process.

To overcome the shortcomings of the two-step DTW-SVMs, and the shortcomings of using a non-PSD kernel in conventional SVMs, and being able to analyse data sets with different lengths in input series, we propose a new approach called GDTW-P-SVMs. It elaborates on P-SVM kernel functions, by utilising the DTW algorithm to provide an elastic distance measure for the kernel function [49]. It utilises GDTW (Equation 5) as the kernel function in potential support vector machines (P-SVMs). In contrast to two-step DTW-SVMs, which calculate the similarities between input series to obtain fixed-length feature vectors (the DTW matrix in Figure 2), GDTW-P-SVMs employ DTW in their kernel to waive the SVM requirement of fixed-length feature vectors.

Potential support vector machines (P-SVMs) have been proposed by Hochreiter and Obermayer to analyse dyadic data where two sets of objects (row and column objects) are characterised by a matrix of numerical values [49]. It is a maximum margin method for construction of classifiers and regression functions for the column objects in a data matrix. The P-SVM optimisation problem can be summarised as follows:

$$\begin{aligned} &\textit{minimise} && \tfrac{1}{2}\|\mathbf{X}_\phi^T\boldsymbol{\omega}\|^2 + C\mathbf{1}^T(\boldsymbol{\xi}^+ + \boldsymbol{\xi}^-) \\ &\textit{subject to} && \mathbf{K}^T(\mathbf{X}_\phi^T\boldsymbol{\omega} - \mathbf{y}) + \boldsymbol{\xi}^+ \geq \mathbf{0} \\ &&& \mathbf{K}^T(\mathbf{X}_\phi^T\boldsymbol{\omega} - \mathbf{y}) - \boldsymbol{\xi}^- \geq \mathbf{0} \\ &&& \mathbf{0} \leq \boldsymbol{\xi}^+, \boldsymbol{\xi}^- \end{aligned} \tag{7}$$

where $\boldsymbol{\omega}$ is a weight vector and $\boldsymbol{\xi}^+$ and $\boldsymbol{\xi}^-$ are slack variables used for the regularisation scheme proposed in [49]. A large value for the slack variables indicates that the particular object only weakly influences the direction of the classification boundary. In Equation 7, $C \geq 0$ is a constant value. If the noise is large, the value of C must be small to remove the corresponding constraints via the slack variables $\boldsymbol{\xi}$. After employing Lagrangian optimisation the following dual optimisation problem will be derived:

$$\begin{aligned} &\textit{minimise} && \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}^T\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}^T\mathbf{K}\boldsymbol{\alpha} \\ &\textit{subject to} && -C\mathbf{1} \leq \boldsymbol{\alpha} \leq C\mathbf{1}, \end{aligned} \tag{8}$$

where $\boldsymbol{\alpha}$ is the vector of Lagrange multipliers. Equation 8 depends on the data via the kernel or data matrix $\mathbf{K}$ only. One of the most crucial properties of the P-SVM procedure is that the dual optimisation problem depends on only $\mathbf{K}$ via $\mathbf{K}^T\mathbf{K}$. Therefore, $\mathbf{K}$ is not required to be PSD or square. This allows the construction of SVM-based classifiers for matrices $\mathbf{K}$ of general shape that includes indefinite kernels. The offset b^* of the classification function $f(\mathbf{x}) = \sum_{i=1}^{l} y_i\alpha_i K(\mathbf{x}_i, \mathbf{x}) + b^*$ is given by [49]:

$$b^* = \frac{1}{l}\sum_{i=1}^{l} y_i. \tag{9}$$

The GDTW-P-SVMs method not only has the advantage of employing the DTW distance measure for comparing input series with different lengths but also overcomes the shortcomings of the two-step DTW-SVMs. As the testing phase for this approach is performed using only the created models, the training set is not required when testing a new data sample. This makes it convenient to distribute the trained data which, are essentially the models. On the other hand, in testing process it is not required to compare each test sample against the entire training set. This makes it an appropriate method for problems that demand real-time classification results.

4. Experiments

To practically evaluate the effectiveness of the two-step DTW-SVMs classifier and GDTW-P-SVMs, we use a set of standard benchmark classification tasks for time series. The experiments used all time series available at the UCR repository [58], the character trajectory data set available at UCI machine learning repository [59] and GeoLife human trajectory data sets [60].

One common method to evaluate classification techniques is comparing results obtained from n-fold cross-validation optimisation. The optimisation involves tuning the hyperparameters (C, γ and kernel parameters) to minimise the error rate. We note that to obtain comparable results, whenever tuning takes place, every adjustment should be considered as a separate independent experiment. The recommended procedure is to use *cross validation tuning* entirely within the training set and use a separate test set for evaluating the classification method [61]. When doing comparative evaluations, everything that is done to modify or prepare the algorithms must be done in advance of seeing the test data [61, 62]. In our experiments, to follow this recommendation a training subset and a testing subset are either pre-defined by the data set providers or a separate tuning set is defined to tune the hyperparameters.

A pairwise strategy with a one-vs-one policy is utilised for multi-class problems. For model selection, a five-fold cross-validation with a leave-one-out policy is performed on each pair of data. In n-fold cross-validation, $n = 5$ and $n = 10$ are the two most commonly used values for the number of folds. In our experiments some data sets have only a few samples in some classes. In these cases the number of samples in a pair can be less than the number of folds and therefore some folds may remain empty. To reduce the frequency of occurrence of empty folds we used $n = 5$ as the number of folds. If the number of samples in a pair is still less than the number of folds, then a sample repetition technique is used to ensure there is at least one sample in each fold. In essence, the sample repetition technique repeats the existing samples with consideration of the balance of data for both classes in the pair.

A shuffling technique is also applied to the data prior to fold generation. The technique helps to maintain a balance of the number of classes in each fold. The accuracy of a model cannot be judged using an unbalanced testing set where the majority of its data belong to one class only. The shuffling technique runs over the data a hundred times to find the folds with balanced training and testing sets.

To perform data classification the LIBSVM [63] and the P-SVM [49] toolboxes are used for implementing two-step DTW-SVMs and GDTW-P-SVMs respectively. To ensure a fair comparison, the hyperparameter selection procedure was equal in all methods. Best values are selected from a generated hyperparameter set to minimise the error rate in the training phase. More precisely, the settings for the GDTW-P-SVMs and the two-step DTW-SVMs are listed below:

- *Two-step DTW-SVMs*: The Gaussian kernel (Equation 2) is used as the kernel function for SVM learning. The best C values (Equation 3) are selected for each fold among a predefined set of values, $\{C_1, C_2, \dots C_{p_c}\}$. We decided to use a logarithmic distribution for C with higher density close to zero. The values in the set are obtained using $C_i = exp(i \times \frac{\ln(C_{max})}{p_c}); i = 0, 1, \dots, p_c$, where p_c and C_{max} are two constant values that indicate number of values and the maximum value for C respectively. The same strategy has been employed for selecting the best γ among $\gamma_i = exp[\ln(\gamma_{min}) + i \times (\frac{\ln(\gamma_{max}) - \ln(\gamma_{min})}{p_\gamma})]; i = 0, 1, \dots, p_\gamma$, where p_γ is the number of values for γ, γ_{min} and γ_{max} are the minimum and maximum values for γ, respectively. In the experiments we choose $C_{\max} = 2^{15}$, $p_c = 2^6$, $\gamma_{\min} = 2^{-10}$, $\gamma_{\max} = 2^3$, $p_\gamma = 2^4$.

- *GDTW-P-SVMs*: We used the GDTW function (Equation 5) as the kernel function for P-SVM classification. C (see Equation 8) and γ ($\gamma = 1/\sigma^2$ in Equation 5) values are selected using the same methods as described for the DTW-SVMs.

All possible permutations of hyperparameters are used to find the minimum classification error rate for $k = 5$ folds. Then for each pair we have $s \geq k$ selected sets (some sets result in the same minimum error rate for the same fold). Among the s sets, the most frequent set with the lowest error rate is determined as the best hyperparameter set for that particular pair. If there are more than one set with that feature then the set that contains the biggest value for C will be selected as the best set. For example assume

$H = \{h_1, \ldots, h_n\}$ is a set that contains all possible permutations of C and γ and it is used to train a classification problem with two classes. For all five folds the error rate associated with each hyperparameter set in H is calculated. For each fold some sets produced the lowest error rate for that fold, these folds are added to set of candidate hyperparameters (S). The most frequent set among all members of S is selected as the best hyperparameter. In case of more than one class of data the same method for choosing the hyperparameters is performed for each pair of classes. This provides a trained model for each pair.

Testing a new data sample is performed against all trained models (one for each pair). A predicted label with the highest number of votes will be selected as the class of the new data sample. If the highest number of votes belongs to more than one class then the same voting algorithm will be performed on the pairs consisting of selected classes only. This routine continues until eventually one class wins the competition. If the voting algorithm fails to find the winner, the label with the lowest class number among selected classes will be chosen as the predicted class for that particular data sample.

The standard DTW algorithm has quadratic time and space complexity that limits its use to only small time series data sets. To overcome this problem, we used the DTW algorithm described in [64]. The algorithm provides the DTW alignments with linear time and space complexity. It uses a multilevel approach that recursively projects a solution from a coarser resolution and refines the projected solution. This makes it possible for the proposed classification technique, GDTW-P-SVMs, to have the same complexity as the P-SVMs with Euclidean distance. As discussed in [65], regardless of the exact algorithm used, the computational cost of solving the SVM Quadratic Problem grows at least like n^2 when C is small and n^3 when C gets large. It depends on number of samples (n), number of support vectors, and the hyperparameters (C and γ).

Large margin classifiers are known to be sensitive to data *normalisation.* The accuracy of a SVM can be severely degraded if the data is not normalized [66]. The main advantage of normalising is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Another advantage is to avoid numerical difficulties during the calculation. Because kernel values usually depend on the inner products of feature vectors, e.g. the linear kernel and the polynomial kernel, large attribute values might cause numerical problems [63]. The normalisation could be performed on input space (on the data sets) and feature space (in the kernel function). The RBF-based kernels normalise the feature space themselves [67]. This does not mean that input space normalisation is not required [66]. In the selected data sets for this experiment, there exist many features. Each of these features may be measured in a different scale and has a different range of possible values. In this case it is beneficial to *scale* all the features to a common range in each data set [67]. This method is also known as standardisation. For scaling our data, a *min-max* method is employed to scale the training data to the common range, $[0, 1]$:

$$\hat{x} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{10}$$

where x_{min} and x_{max} are minimum and maximum values of the scaling data set, $\hat{x}$ is the scaled data, and x is the raw data sample. As previously discussed in section 2.3, the DTW-SVMs classification technique requires the training set as well as the trained models to test data samples. Maximum and minimum values of the training set are used to scale the test set to the desired range.

The proposed classification method is examined with two types of data sets:

1. *Fixed-length feature vectors*: Data sets where the data have a certain number of features and the values of all features for all samples are provided.
2. *Variable-length input series*: Data sets where the data do not have a fixed number features or values of some features for some samples are not available. Trajectory-based data sets are one of the most common examples of this type, such as character trajectory data sets and human trajectory data sets.

The next two subsections discuss classification results obtained using both types of data.

4.1. Fixed-Length Feature Vector Classification

This section describes experiments that compare various classification techniques using data sets with fixed-length feature vectors. The classification performance of GDTW-P-SVMs is compared with two groups of classifiers. The first group contains classifiers that use the first nearest neighbour (1NN) technique along with a selection of common distance measures to classify UCR data sets. In the second group we compare our method with kernel-based classifiers that employ DTW as a distance measure in their kernel. The outcome is a pairwise comparison of these classifiers with respect to their classification accuracy.

4.1.1. UCR data sets

The UCR time series data sets [58] are used to benchmark our proposed methods and compare them with some other previously presented classification techniques. An overview of the UCR data sets and their properties is given in Table 1. This benchmark database includes a wide variety of practical classification problems including speech recognition, face recognition, motion tracking data analysis, and electrocardiography data classification. The length of the time series varies from 60 to 637 time steps and the data sets contain 24,009 time series in total. Each of the 20 data sets comes with a training set and a test set.

Table 2 shows the results obtained for different classification methods. The methods, which are discussed in this table, employed the 1-nearest neighbor classifier with a distance measure. We use the 1-nearest neighbor classifier because the 1-nearest neighbor classifier with DTW showed very competitive performance and has been widely used for time series classification [24]. In the table, for space concerns, the acronyms of the methods' names are used: 1NN ED (first nearest neighbour with Euclidean Distance), DTW (classic Dynamic Time Warping [21]), ODTW (Optimised Dynamic Time Warping [68]), LCSS (Longest Common Sub-sequence [22]) and ERP (Edit distance with Real Penalty [23]).

Figure 5 compares our proposed classification technique with the other methods. Red dots and blue dots show error rates for GDTW-P-SVMs and DTW-SVMs, respectively for each data set. The black line in each diagram is representative of the case where both methods undergoing comparison would have equal error rates. More blue/red dots above the black line means the DTW-SVMs/GDTW-P-SVMs have lower classification error rates than the comparing method. As shown the GDTW-P-SVMs (red dots) have lower error rates in most cases even when comparing with powerful distance measures such as LCSS and ERP (last two diagrams in Figure 5). The DTW-SVMs have a lower error rate than most other techniques, but they always have a higher error rate than GDTW-P-SVMs.

Figure 6 shows Receiver Operating Characteristic (ROC) curves of the GDTW-P-SVMs and SVM with ED-Gaussian Kernel classifiers for the five data sets from the UCR repository which have two classes (binary classifica-

Table 1: UCR time series data set properties [58]

Dataset Name	Number of classes	Size of training set	Size of testing set	Length of series
Synthetic control	6	300	300	60
Gun-Point	2	50	150	150
CBF	3	30	900	128
Face(all)	14	560	1690	131
OSU Leaf	6	200	242	427
Swedish Leaf	15	500	625	128
50 Words	50	450	455	270
Trace	4	100	100	275
Two Patterns	4	1000	4000	128
Wafer	2	1000	6174	152
Face(four)	4	24	88	350
Lightning-2	2	60	61	637
Lightning-7	7	70	73	319
ECG	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
Olive oil	4	30	30	570

Table 2: UCR time series classification error rates, 1NN: first nearest neighbour, ED: Euclidean distance, ODTW: optimised DTW, LCSS: longest common sub-sequence, ERP: edit distance with real penalty. GDTW-P-SVMs show better results for automatic time series classification with fixed-length feature vectors. Best result(s), i.e. lowest error rate, for each data set are shown in bold.

Dataset Name	1NN ED	1NN ODTW	1NN DTW	1NN LCSS	1NN EPR	DTW SVM	GDTW P-SVM
Synthetic control	0.12	0.017	0.007	0.047	0.036	0.007	**0.000**
Gun-Point	0.087	0.087	0.093	0.013	0.040	0.200	**0.000**
CBF	0.148	0.004	0.003	0.009	0.003	**0.000**	**0.000**
Face (all)	0.286	0.192	0.192	0.201	0.201	0.256	**0.102**
OSU Leaf	0.483	0.384	0.409	**0.202**	0.397	0.355	0.330
Swedish Leaf	0.213	0.157	0.210	0.117	0.120	0.184	**0.094**
50 Words	0.369	0.242	0.310	**0.213**	0.281	0.264	0.222
Trace	0.24	0.01	**0.000**	0.20	0.170	**0.000**	**0.000**
Two Patterns	0.090	0.0015	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
Wafer	0.005	0.005	0.020	**0.000**	0.009	0.010	**0.000**
Face (four)	0.216	0.114	0.170	0.068	0.102	0.079	**0.023**
Lightning-2	0.246	**0.131**	**0.131**	0.180	0.148	0.197	0.164
Lightning-7	0.425	0.288	0.274	0.452	0.301	0.370	**0.260**
ECG	0.120	0.120	0.230	**0.100**	0.130	0.150	**0.100**
Adiac	0.389	0.391	0.396	0.452	0.378	0.371	**0.289**
Yoga	0.170	0.155	0.164	**0.137**	0.147	0.151	0.147
Fish	0.217	0.233	0.267	**0.091**	0.120	0.206	0.194
Beef	**0.467**	**0.467**	0.500	0.533	0.500	0.500	0.500
Coffee	0.250	0.179	0.179	0.214	0.250	0.179	**0.000**
Olive oil	**0.133**	0.167	**0.133**	0.800	0.167	**0.133**	**0.133**
Average Rank	4.650	3.000	3.650	2.800	3.000	3.100	1.400

tion problems) [69]. The False Positive Rate (FPR) is defined as the fraction of the false negatives out of total negatives, and the True Positive Rate (TPR) is defined as the fraction of the true positives out of total positives. The Area Under Curve (AUC) when using GDTW-P-SVMs for ECG, GunPoint, Wafer, Coffee, and Yoga data sets are 0.825, 0.985, 0.916, 1.000, and 0.894, respectively, and when using SVMs they are 0.891, 0.837, 0.688, 0.713, and 0.568[3]. The ROC curves show that GTDW-P-SVMs have higher accuracy in classifying positive and negative samples than SVM with Gaussian kernel. They also support the classification error rates presented in Table 2.

As seen in Figure 5 and in the last column of Table 2, GDTW-P-SVMs clearly outperform the other classification methods; in most cases the accuracy of GDTW-P-SVMs is higher than that of others. The experimental results for fixed-length feature vectors indicate that our proposed method (GDTW-P-SVMs) is promising for automatic time series classifications with fixed-length feature vectors.

Table 3 compares classification results obtained using kernel-based classification techniques which use DTW as the distance measure in their kernel function. In the table, for space concerns, the acronyms of the methods' names are used: ppfSVM-NDTW (pairwise proximity function SVM [70] with negated DTW kernel [57]), ppfSVM-GDTW (pairwise proximity function SVM with GDTW kernel), SVM-NDTW (conventional SVM with negated DTW kernel), SVM-GDTW (conventional SVM with GDTW kernel).

Figure 7 compares our proposed classification technique with other methods, which were discussed in Table 3. In this figure, we used the same representation as in Figure 5. As shown the GDTW-P-SVMs (red dots) have lower error rates in all cases even when comparing them with pairwise proximity function SVMs with GDTW and NDTW kernels (last two diagrams in Figure 5).

The last row of Table 2 and Table 3 shows the *average rank* of each classifier using the Friedman test [71]. The average rank is calculated for each group of classifiers separately. To obtain the average rank initially the classifiers were ranked on each data set separately. Then for each data set the classifier with lowest error rate (highest performance) is assigned rank 1, the second best rank 2, and so on. In the case of ties average ranks are

---

[3] The closer the value of AUC to 1, higher the accuracy of classification.

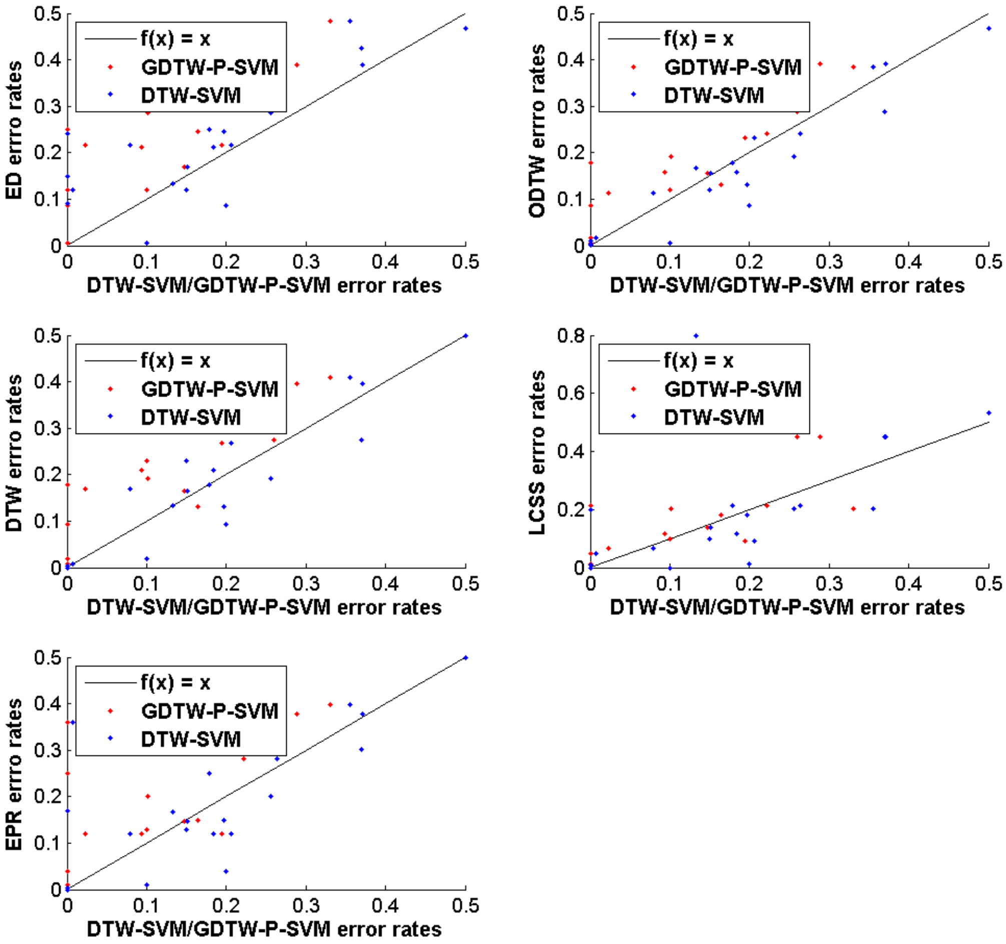

Figure 5: Comparison of time series classification methods (presentation method adopted from [18]). $x-axes$ represent error rates for DTW-SVMs and GDTW-P-SVMs with blue and red dots respectively. $y-axes$ show error rates for the other five classification methods which were compared in Table 2. Black lines represent $f(x) = x$. More blue/red dots above the black line means that the DTW-SVMs/GDTW-P-SVMs have lower classification error rates than the comparing method. As shown the GDTW-P-SVMs (red dots) have lower error rates in most cases even when compared with powerful distance measures such as LCSS and ERP (last two diagrams in the figure).

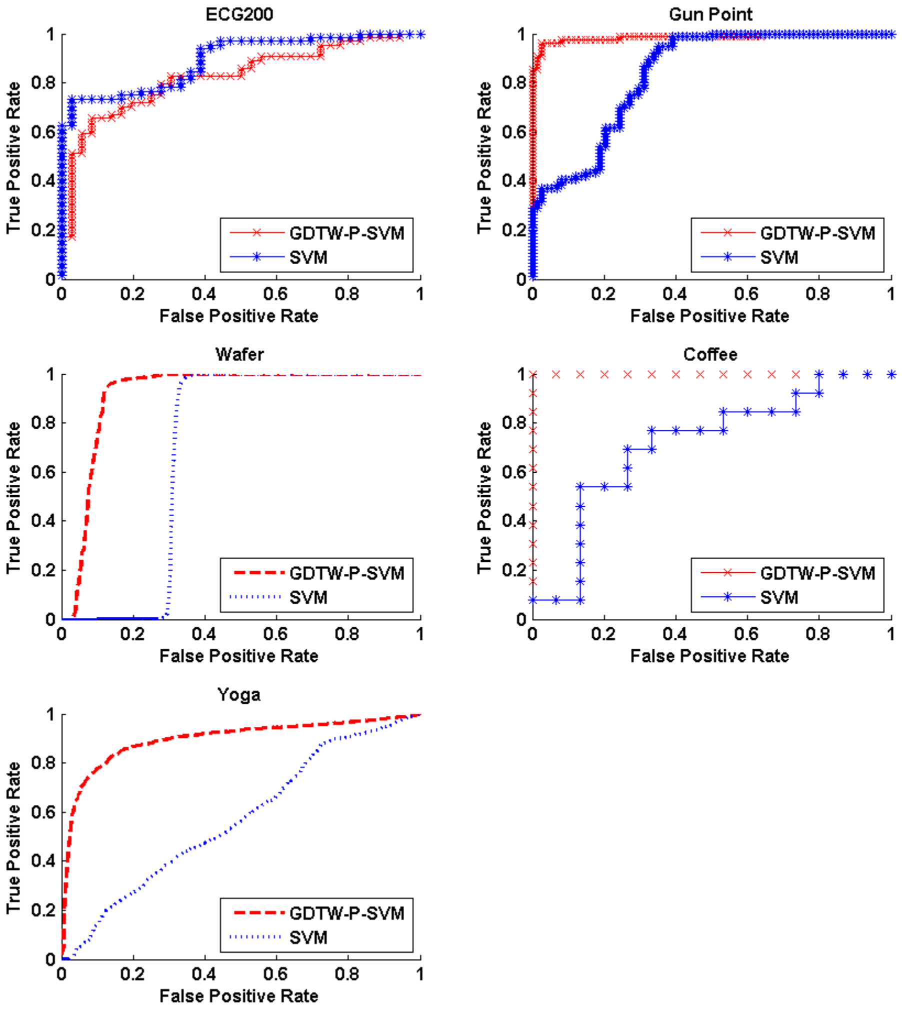

Figure 6: Comparison of Receiver Operating Characteristic (ROC) curves for SVMs with Gaussian Kernel and GDTW-P-SVMs for five UCR data sets with two classes; False Positive Rate (FPR) is defined as the fraction of the false negatives out of total negatives. True Positive Rate (TPR) is defined as the fraction of the true positives out of total positives. The Area Under Curve (AUC) when using GDTW-P-SVMs for ECG, Gun-Point, Wafer, Coffee, and Yoga data sets are 0.825, 0.985, 0.916, 1.000, and 0.894, respectively, and when using SVM they are 0.891, 0.837, 0.688, 0.713, and 0.568 (the closer the value of the AUC is to 1, the higher is the accuracy of classification).

Table 3: UCR time series classification error rates using DTW-based kernel function classifiers. ppfSVM/NDTW: pairwise proximity function SVM with negated DTW kernel, ppfSVM/GDTW: pairwise proximity function SVM with GDTW kernel, SVM/NDTW: conventional SVM with negated DTW kernel, SVM/GDTW: conventional SVM with GDTW kernel. GDTW-P-SVMs show promising results for automatic time series classification with fixed-length feature vectors. Best result(s) for each data set is shown in bold. The classifiers were ranked on each data set according to their performance and the ranks averaged over all data sets (lower rank indicates better performance.)

Dataset Name	ppfSVM NDTW	ppfSVM GDTW	SVM NDTW	SVM GDTW	DTW SVM	GDTW P-SVM
Synthetic control	0.013	0.013	0.013	0.023	0.007	**0.000**
Gun-Point	0.047	0.140	0.460	0.127	0.200	**0.000**
CBF	0.003	0.001	0.010	0.046	**0.000**	**0.000**
Face(all)	0.237	0.226	0.170	0.265	0.256	**0.102**
OSU Leaf	0.405	0.355	0.706	0.401	0.355	**0.330**
Swedish Leaf	0.147	0.155	0.363	0.382	0.184	**0.094**
Trace	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
Two Patterns	**0.000**	0.001	0.007	**0.000**	**0.000**	**0.000**
Wafer	0.010	0.015	0.181	0.034	0.010	**0.000**
Face(four)	0.148	0.114	0.102	0.114	0.079	**0.023**
Lightning-2	0.328	0.316	0.492	**0.115**	0.197	0.164
Lightning-7	0.301	0.315	0.219	0.301	0.370	**0.260**
ECG	0.160	0.220	0.440	0.170	0.150	**0.100**
Adiac	0.325	0.343	0.512	0.560	0.371	**0.289**
Yoga	0.227	0.177	0.534	0.219	0.151	**0.147**
Fish	**0.189**	0.240	0.240	0.297	0.206	0.194
Beef	0.567	0.533	0.633	0.600	**0.500**	**0.500**
Coffee	0.107	0.179	0.500	0.179	0.179	**0.000**
Olive oil	0.167	0.267	**0.133**	0.267	**0.133**	**0.133**
Average rank	3.421	3.868	4.737	4.500	3.052	1.421

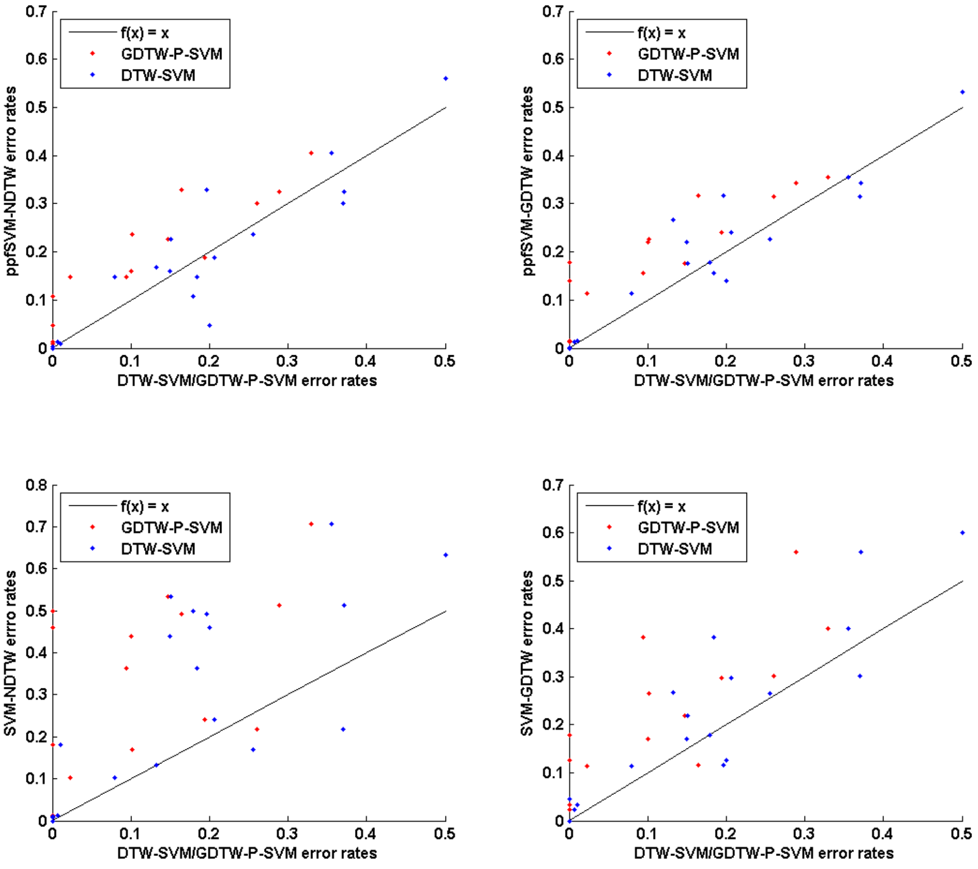

Figure 7: Comparison of time series classification methods (presentation method adopted from [18]). $x-axes$ represent error rates for DTW-SVMs and GDTW-P-SVMs with blue and red dots respectively. $y-axes$ show error rates for the other four classification methods which were compared in Table 3. Black lines represent $f(x) = x$. More blue/red dots above the black line means that the DTW-SVMs/GDTW-P-SVMs have lower classification error rates than the comparing method. As shown the GDTW-P-SVMs (red dots) have lower error rates in most cases even when compared with ppfSVM-NDTW and ppfSVM-GDTW (the first two diagrams in the figure).

assigned for that data set. Then the ranks are averaged over all data sets in each group.

Critical value for the two-tailed Bonferroni-Dunn test [72] with $\alpha = 0.05$ for Table 2 is $q_\alpha = 2.638$. Critical difference (CD)[4] for this table is obtained as:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} = 1.8021 \tag{11}$$

Where k is the number of classification techniques and N is the number of data sets. Critical value for the two-tailed Bonferroni-Dunn test with $\alpha = 0.05$ for Table 3 is $q_\alpha = 2.576$. Critical difference (CD) is obtained as:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} = 1.5636 \tag{12}$$

A pairwise comparison of the average rank of classifiers and the obtained critical value for Table 2 and Table 3 using the Bonferroni-Dunn test are presented in Table 4 and Table 5, respectively. Bold values in Table 4 and 5 show that the corresponding classifier on left performs significantly better than the corresponding classifier on top. The values in these two tables are obtained by subtracting the average ranks of corresponding classifiers. If the value is more than the critical difference then the difference between the compared classifiers is significant [73].

The differences between the rank of GDTW-P-SVMs and the ranks of other the classification methods, in the first group of classifiers, are always greater than the CD (obtained in Equation 11). Therefore GDTW-P-SVMs perform significantly better than the other classification methods that are discussed in Table 2. Although the difference between GDTW-P-SVMs and 1NN-LCSS is just above the CD, still GDTW-P-SVMs have statistically significantly better performance. In the second group of compared classifiers, the rank of GDTW-P-SVMs showed even a greater difference compared to others. As shown in Table 5, the differences between the rank of GDTW-P-SVMs and the ranks of the other classification methods are always greater than the CD obtained in Equation 12. Therefore GDTW-P-SVMs perform significantly better than the other classification methods that are shown in Table 3.

---

[4] The performance of two classifiers is significantly different if the corresponding average ranks differ by at least the critical difference [73].

Table 4: A pairwise comparison of the average rank of classifiers discussed in Table 2 using Bonferroni-Dunn test. The classifiers employed 1-NN with a selection of common measure distances. Bold values in the table show that the corresponding classifier on left performs significantly better than the corresponding classifier on top. The $CD = 1.802$ for this group of classifiers is obtained using Equation 11.

Classifiers	1NN ED	1NN ODTW	1NN DTW	1NN LCSS	1NN EPR	DTW SVM	GDTW P-SVM
1NN NDTW	N/A	-1.75	-1.225	-2	-1.65	-1.625	-3.825
1NN ODTW	1.75	N/A	0.525	-0.25	0.1	0.125	-2.075
1NN DTW	1.225	-0.525	N/A	-0.775	-0.425	-0.4	-2.6
1NN LCSS	2	0.25	0.775	N/A	0.35	0.375	-1.825
1NN EPR	1.65	-0.1	0.425	-0.35	N/A	0.025	-2.175
DTW SVM	1.625	-0.125	0.4	-0.375	-0.025	N/A	-2.2
GDTW P-SVM	**3.825**	**2.075**	**2.6**	**1.825**	**2.175**	**2.2**	N/A

Table 5: A pairwise comparison of the average rank of classifiers with DTW-based kernel (discussed in Table 3) using Bonferroni-Dunn test. Bold values in the table show that the corresponding classifier on left performs significantly better than the corresponding classifier on top. The $CD = 1.5636$ for this group of classifiers is obtained using Equation 12.

Classifiers	ppfSVM NDTW	ppfSVM GDTW	SVM NDTW	SVM GDTW	DTW SVM	GDTW P-SVM
ppfSVM NDTW	N/A	0.447	1.316	1.079	-0.368	-2
ppdfSVM GDTW	-0.447	N/A	0.868	0.631	-0.816	2.447
SVM NDTW	-1.316	-0.868	N/A	-0.237	-1.684	-3.316
SVM GDTW	-1.079	-0.631	0.237	N/A	-1.447	-3.079
DTW SVM	0.368	0.816	1.684	1.447	N/A	-1.632
GDTW P-SVM	**2**	**2.447**	**3.316**	**3.079**	**1.632**	N/A

We have not used the ANOVA [74] to evaluate our classifier because: 1. ANOVA assumes that the classification error rates (performance) are drawn from a normal distribution, which is not always the case in general. 2. ANOVA requires that random variables have equal variance. Neither learning algorithms, nor data sets can satisfy this condition [73].

In the next two subsections the proposed classification method is tested against two trajectory-based data sets with variable-length input series.

4.2. Variable-length feature series classification

The comprehension of phenomena related to movement – not only of people and vehicles but also of animals and other moving objects – has always been a key issue in many areas of scientific investigation and social analysis. Data collected for movement based analysis are called *trajectory data* and it can be represented as sequences of time stamped locations. Trajectory data are normally obtained from location-aware devices that capture the position of an object during a specific time interval. Since object movements can occur at different speeds the trajectory data are variable-length input series, which makes them suitable data sets for the GDTW-P-SVMs. In this section we present our classification results for the character [59] and human [75] trajectory data sets.

4.2.1. Character trajectory data set

The character trajectory data set consists of labelled samples of pen tip trajectories recorded whilst writing individual characters. All samples are from the same writer, for the purposes of primitive extraction. Only characters with a single pen-down segment were considered. The data consist of 2858 character samples with different lengths. Each sample is a 3-dimensional velocity trajectory (x, y, and pen tip force). The data has been numerically differentiated and Gaussian smoothed, with a sigma value of 2 [59, 76]. The classification task is to recognise characters in the data set using trained models.

Table 6 presents the classification error rates resulting from our experiments. Three data representations are used for character classification in [77]:

1. *Likelihood*: Employs the label information that is available for the objects in the training data and represents the data using maximum likelihood [77].

Table 6: Classification error rate (%) on the handwritten character data set for four different classification methods presented in [77] as well as our classification methods. The data set includes data objects with different lengths input series. While no feature presentation method is applied on the data for GDTW-P-SVMs and DTW-SVMs, they have shown the lowest classification error rates.

Classifier	Feature Representation	Error rate
Bayes	Likelihood	12.46
Softmax	Likelihood	8.14
	Fisher	8.23
	Fisher Kernel Learning	6.95
SVMs	Likelihood	7.91
	Fisher	7.64
	Fisher Kernel Learning	6.91
DTW-SVM	–	**5.450**
GDTW-P-SVM	–	**3.010**

2. *Fisher kernel*: The Fisher kernel is defined as the inner product of the directions of gradient ascent, i.e., the inner product of the natural gradients. It simply uses the gradients as features, without any further rescalings or normalizations [76].
3. *Fisher kernel learning (FKL)*: Trains the model in such a way that objects with the same class induce gradients that are similar, whereas objects with different classes induce log-likelihood gradients that are dissimilar [77].

In our proposed classification method, we used a simple data projection for representing the data. It projects 3D trajectory data (2D coordinates and pen force value) into 1D variable-length sequential data samples. As seen in Table 6 the proposed method has lower error rates compared to the other methods.

4.2.2. Human trajectory data sets

The rise of GPS and broadband-speed wireless devices has led to a range of applications broadly characterized as location based services. These applications will provide users with information that is targeted and personalized to their location, whether it be nearby stores, friends, traffic conditions, etc.

The human trajectory data set that we used in our experiments was col-

Table 7: Total distance and duration of transportation modes in the GeoLife data set [75].

Transportation Mode	Distance (km)	Duration (hours)	#train	#test
Walk	11,457	5,126	1,586	1,910
Bike	6,335	2,304	274	602
Bus	21,931	1,430	930	629
Car and Taxi	34,127	2,349	318	324
Train	74,449	459	412	870
Total	18,7679	12,041	3,520	4,335

lected in the Geolife project by 167 users in a period of over three years. A GPS trajectory of this data set is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude and altitude. This data set contains 17,355 trajectories with a total distance of about 1 million kilometers and a total duration of 48,000+ hours. These trajectories were recorded by different GPS loggers and GPS-phones. This data set recorded a broad range of users' outdoor movements, including not only life routines like going home and going to work but also some entertainment and sports activities, such as shopping, sightseeing, dining, hiking, and cycling [75].

The classification task defined here is based on supervised learning to automatically recognise users' transportation modes, such as driving, walking, taking a bus, riding a bike and traveling on a train, from raw GPS logs. 59 users have labeled their trajectories with transportation mode. The total distance and duration of transportation modes are listed in Table 7. Trajectories with unknown and airplane transportation mode were excluded from the data set.

Table 8 shows the classification accuracy for our approaches as well as four other classification methods described in [78] over the training set and testing set. In [78] two segmentation methods, by length and by duration, along with a classifier were used to recognise the transportation mode. As our proposed method is able to handle data samples with different lengths this step can be waived and the raw GPS trajectory data can be used to recognise the transportation mode. Here again a dimension projection that projects 3D (latitude, longitude, and altitude) into 1D variable-length sequential data is applied. As seen in Table 8 the accuracy of our proposed approach (GTDW-

Table 8: Classification accuracy for the GeoLife data set. The data set includes data objects with variable-length input series. While no segmentation method is applied to the data for GDTW-P-SVMs and DTW-SVMs methods, they have shown the highest classification accuracy.

Classifier	Segmentation Method	Accuracy (%)
Decision Tree	by length	70
	by duration	75
SVMs	by length	57
	by duration	62
Bayes net	by length	69
	by duration	71
CRF	by length	53
	by duration	40
DTW-SVM	–	**79**
GDTW-P-SVMs	–	**81**

PSVMs) is higher than the other approaches.

5. Discussion and Future Work

The new technique presented in this article coupled a SVM-based classification technique with an indefinite kernel. We compared our coupling combination with a number of other combinations that have been recently proposed (SVMs-NDTW, SVMs-GDTW, ppfSVMs-NDTW, and ppfSVMs-GDTW). In addition to those combinations, there exist a variety of indefinite kernels and classification techniques that can be coupled [39, 40]. The kernels do not satisfy Mercer's condition and they induce associated functional spaces called Reproducing Kernel Krein Spaces (RKKS), which are a generalisation of Reproducing Kernel Hilbert Spaces (RKHS). This article emphasised the importance of such couplings by giving GDTW-P-SVMs as an example with a classification accuracy that is significantly higher than existing methods for wide varieties of benchmarked data sets. Experimenting with indefinite kernels and other combinations of kernel-based classification techniques that can handle indefinite kernels is a possible direction of future research.

Although in the experiments we employed GDTW-P-SVMs to solve classification problems, the ability of GDTW-P-SVMs to handle variable length data objects can be utilised for time series segmentation. For example, an energy-based model for unsupervised factorisation has been employed for unsupervised time series segmentation with fixed-length using SVMs [79, 80]. A similar approach could be applicable for segmenting time series with variable length using GDTW-P-SVMs.

6. Conclusion

We introduced a new classification technique, GDTW-P-SVMs, for sequential data analysis where each data object is characterised by a series of numerical values that may have different lengths for different data objects. The well-known DTW algorithm was utilised to provide an elastic distance measure that is able to compare variable-length input series. We compared GDTW-P-SVMs with the two-step DTW-SVMs method where training data were required in the testing phase as well as the trained models. Although DTW-SVMs were able to classify trajectory data with acceptable error rates, they are not able to provide the classification results in real-time as the testing phase for this technique is too slow for problems that have a big training set. GDTW-P-SVMs were proposed to overcome the shortcomings of the DTW-SVMs by altering the kernel function in P-SVMs using DTW. As a result, GDTW-P-SVMs could handle data and kernel matrices that were neither positive definite nor square, and it could also be applied to data with variable-length input series. Benchmarks for classification were performed with several real-world data sets from the UCR Time Series Classification/Clustering page, the GeoLife trajectory data set, and the character trajectory from the UCI repository. The data sets included data with both variable and fixed-length input series. In the case of variable-length data samples, GDTW-P-SVMs significantly outperformed other existing methods by two main advantages: *i*) the proposed method had significantly lower classification error rates and *ii*) it waived the need for data representation as fixed-length feature vectors. The second advantage is important when the extraction of fixed-length feature vectors is not feasible or when using fixed-length segments of data objects fails to describe the relationship between data objects properly. We are currently investigating possible applications of the proposed classification method.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful comments. This project is supported by ARC DP1092679 “Modelling and

predicting patterns of pedestrian movement: using robotics and machine learning to improve the design of urban space".

References

[1] B. Wang, H. Huang, X. Wang, A novel text mining approach to financial time series forecasting, Neurocomputing 83 (1) (2012) 136 – 145.

[2] H. Wu, B. Salzberg, D. Zhang, Online event-driven subsequence matching over financial data streams, in: G. Weikum, A. C. Konig, S. Debloch (Eds.), Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, SIGMOD'04, ACM, New York, NY, USA, 2004, pp. 23–34.

[3] Z. Liao, J. Wang, Forecasting model of global stock index by stochastic time effective neural network, Expert Systems with Applications 37 (1) (2010) 834–841.

[4] A. Jalalian, M. Fathy, Pedestrian detection from a moving camera with an advanced camera-motion estimator, in: K. Yetongnon, R. Chbeir, A. Dipanda (Eds.), 3rd International IEEE Conference on Signal-Image Technologies and Internet-Based System, IEEE Computer Society, Washington, DC, USA, 2007, pp. 965–971.

[5] Z. Xue, D. Ming, W. Song, B. Wan, S. Jin, Infrared gait recognition based on wavelet transform and support vector machine, Pattern Recognition 43 (8) (2010) 2904–2910.

[6] T. Vincent, L. Risser, P. Ciuciu, Spatially adaptive mixture modeling for analysis of fmri time series, IEEE Transactions on Medical Imaging 29 (4) (2010) 1059–1074.

[7] D. R. Lowne, S. J. Roberts, R. Garnett, Sequential non-stationary dynamic classification with sparse feedback, Pattern Recognition 43 (3) (2010) 897–905.

[8] M. Cuturi, J.-P. Vert, A mutual information kernel for sequences, in: Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, Vol. 3, IEEE Computer Society, Washington, DC, USA, 2004, pp. 1905–1910.

[9] A. Jalalian, S. K. Chalup, M. J. Ostwald, Intelligent evaluation of urban streetscape designs by analysing pedestrian body dynamics, in: The Third International Workshop on Advanced Computational Intelligence (IWACI2010), IEEE Computer Society, Washington, DC, USA, 2010, pp. 442–447.

[10] A. Jalalian, S. K. Chalup, M. J. Ostwald, Agent-agent interaction as a component of agent-environment interaction in the modelling and analysis of pedestrian visual behaviour, in: C. M. Herr, N. Gu, S. Roudavski, M. A. Schnabel (Eds.), The 16th International Conference of the Association for Computer-Aided Architectural Design Research in Asia, Newcastle, Australia, 2011, pp. 555–564.

[11] A. Jalalian, S. K. Chalup, M. J. Ostwald, Simulating pedestrian flow dynamics for evaluating the design of urban and architectural space, in: C. Murphy, S. J. Wake, D. Turner, G. McConchie, D. Rhodes (Eds.), 44th Annual Conference of the Architectural Science Association, ANZAScA 2010, Unitec Institute of Technology, Auckland, New Zealand, 2010.

[12] A. Jalalian, S. K. Chalup, M. J. Ostwald, Architectural evaluation of simulated pedestrian spatial behaviour, Architectural Science Review 54 (2) (2011) 132–140.

[13] Y. Yasami, S. Khosravi, S. Pour Mozaffari, A. Jalalian, An unsupervised network anomaly detection approach by k-means clustering and id3 algorithms, in: Symposium on Computers and Communications, IEEE Computer Society, Washington, DC, USA, 2008, pp. 398–403.

[14] N. G. Pavlidis, D. K. Tasoulis, N. M. Adams, D. J. Hand, Lambda-perceptron: An adaptive classifier for data streams, Pattern Recognition 44 (1) (2011) 78–96.

[15] Y. W. Huang, P. S. Yu, Adaptive query processing for time-series data, in: S. Chaudhuri, D. Madigan, U. Fayya (Eds.), Proceedings of the 5th Int'l Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 1999, pp. 282–286.

[16] P. Indyk, N. Koudas, S. Muthukrishnan, Identifying representative trends in massive time series data sets using sketches, in: A. E. Abbadi,

M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, K.-Y. Whang (Eds.), Proceedings of the 26th International Conference on Very Large Data Bases, VLDB'00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 363–372.

[17] K. Kalpakis, D. Gada, P. Vasundhara, Distance measures for effective clustering of arima time-series, in: N. Cercone, T. Y. Lin, X. Wu (Eds.), Proceedings of the 2001 IEEE International Conference on Data Mining, IEEE Computer Society, Washington, DC, USA, 2001, pp. 273–280.

[18] P.-F. Marteau, Time warp edit distance with stiffness adjustment for time series matching, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2009) 306–318.

[19] E. Deza, M. M. Deza, Encyclopedia of distances, Springer-Verlag Berlin Heidelberg, New York, USA, 2009.

[20] R. Courant, D. Hilbert, Methods of mathematical physics, first english Edition, Vol. 1, Interscience Publishers, New York, USA, 1937, translated and revised from the German original.

[21] V. M. Velichko, N. G. Zagoruyko, Automatic recognition of 200 words, International Journal of Man-Machine Studies 2 (3) (1970) 223–234.

[22] V. Chvatal, D. Sankoff, Longest common subsequences of two random sequences, Journal of Applied Probability 12 (2) (1975) 306–315.

[23] L. Chen, N. Raymond, On the marriage of lp-norms and edit distance, in: M. A. Nascimento, M. T. Ozsu, D. Kossmann, J. A. B. Renee J. Miller, B. Schiefer (Eds.), Proceedings of the 30th Very Large Databases (VLDB) Conference, Morgan Kaufmann Publishers, San Francisco, CA, USA, 2004, pp. 792–803.

[24] Y.-S. Jeong, M. K. Jeong, O. A. Omitaomud, Weighted dynamic time warping for time series classification, Pattern Recognition 44 (9) (2011) 2231–2240.

[25] E. J. Keogh, M. J. Pazzani, Derivative dynamic time warping, in: R. Grossman, V. Kumar, J. Han (Eds.), First SIAM International Conference on Data Mining, SIAM, Philadelphia, PA 19104-2688 USA, 2001, pp. 1–11.

[26] C. A. Ratanamahatana, E. Keogh, Three myths about dynamic time warping data mining, in: Proceedings of the 2005 SIAM International Conference on Data Mining, 2005, pp. 506–510.

[27] E. Keogh, S. Kasetty, On the need for time series data mining benchmarks: a survey and empirical demonstration, in: The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'02, ACM, New York, NY, USA, 2002, pp. 102–111.

[28] S. Park, S.-w. Kim, W. W. Chu, Segment-based approach for subsequence searches in sequence database, in: Proceedings of the 2001 ACM Symposium on Applied Computing, SAC'01, ACM, New York, NY, USA, 2001, pp. 248–252.

[29] C. Wang, X. S. Wang, Supporting content-based searches on time series via approximation, in: O. Gunther, H. J. Lenz (Eds.), The 12th International Conference on Scientific and Statistical Database Management, IEEE Computer Society, Washington, DC, USA, 2000, pp. 69–81.

[30] C. Orsenigo, C. Vercellis, Combining discrete svm and fixed cardinality warping distances for multivariate time series classification, Pattern Recognition 43 (11) (2010) 3787–3794.

[31] M. Wöllmer, M. Al-Hames, F. Eyben, B. Schuller, G. Rigoll, A multidimensional dynamic time warping algorithm for efficient multimodal fusion of asynchronous data streams, Neurocomputing 73 (1) (2009) 366–380.

[32] C. Ferrer, D. Torres, M. E. Hernández-Díaz, Using dynamic time warping of t0 contours in the evaluation of cycle-to-cycle pitch detection algorithms, Pattern Recognition Letters 31 (6) (2011) 517–522.

[33] E. J. Keogh, M. J. Pazzani, Scaling up dynamic time warping for datamining applications, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'00, ACM, New York, NY, USA, 2000, pp. 285–289.

[34] B. Hartmann, N. Link, Gesture recognition with inertial sensors and optimized dtw prototypes, in: Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, IEEE Computer Society, Washington, DC, USA, 2010, pp. 2102–2109.

[35] J. Shawe-Taylor, S. Sun, A review of optimization methodologies in support vector machines, Neurocomputing 74 (17) (2011) 3609 – 3618.

[36] V. N. Vapnik, Statistical learning theory, Adaptive and Learning Systems for Signal Processing, Communications, and Control, Wiley, New York, NY, USA, 1998.

[37] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (7) (2001) 1443–1471.

[38] Z. Liang, Y. Li, Incremental support vector machine learning in the primal and applications, Neurocomputing 72 (10) (2009) 2249 – 2258.

[39] C. S. Ong, X. Mary, S. Canu, A. J. Smola, Learning with non-positive kernels, in: Proceedings of the Twenty-First International Conference on Machine Learning, ICML '04, ACM, New York, NY, USA, 2004, pp. 81–89.

[40] R. Luss, A. d'Aspremont, Support vector machine classification with indefinite kernels, Computing Research Repository, CoRR abs/0804.0188 (2009) 1–8.

[41] M. Seeger, Covariance kernels from bayesian generative models, in: S. T. S. Becker, K. Obermayer (Eds.), Neural Information Processing Systems, Vol. 14, The MIT Press, Cambridge MA. USA, 2000, pp. 905–912.

[42] N. Cristianini, J. Shawe-Taylor, An introduction to support vector machines and other kernel-based learning methods, tenth Edition, Cambridge University Press, Cambridge CB2 2RU, UK, 2006.

[43] H. W. Kuhn, A. W. Tucker, Nonlinear programming, in: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, CA, USA, 1951, pp. 481–492.

[44] M. Marcus, H. Minc, Introduction to linear algebra, Dover Publications, New York, USA, 1988.

[45] J. Mercer, Functions of positive and negative type, and their connection with the theory of integral equations, Philosophical Transactions of the

Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 209 (1909) 415–446.

[46] F. Riesz, B. Szökefalvi Nagy, Functional analysis, Dover Publications, INC., Mineola, N.Y., USA, 1990.

[47] M. M. Adankon, M. Cheriet, Optimizing resources in model selection for support vector machine, Pattern Recognition 40 (3) (2007) 953–963.

[48] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, K. Obermayer, Classification on pairwise proximity data, in: M. S. Kearns, S. A. Solla, D. A. Cohn (Eds.), Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems 11, MIT Press, Cambridge, MA, USA, 1999, pp. 438–444.

[49] S. Hochreiter, K. Obermayer, Support vector machines for dyadic data, Neural Computation 18 (6) (2006) 1472–1510.

[50] I. J. Schoenberg, Metric spaces and positive definite functions, Transactions of the American Mathematical Society 44 (3) (1938) 522–536.

[51] C. Bahlmann, B. Haasdonk, H. Burkhardt, On-line handwriting recognition with support vector machines - a kernel approach, in: A. D. Williams (Ed.), The 8th International Workshop on Frontiers in Handwriting, IWFHR'02, IEEE Computer Society, Washington, DC, USA, 2002, pp. 49–54.

[52] H. Shimodaira, K.-i. Noma, M. Nakai, S. Shigeki, Dynamic time-alignment kernel in support vector machine, in: T. G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14, MIT Press, Cambridge, MA, USA, 2002, pp. 921–928.

[53] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics, Speech and Signal Processing 26 (1) (1978) 43–49.

[54] H. Lei, B. Sun, A study on the dynamic time warping in kernel machines, in: K. Yetongnon, R. Chbeir, A. Dipanda (Eds.), Signal-image technologies and internet-based systems, IEEE Computer Society, Washington, DC, USA, 2007, pp. 839–845.

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

[56] J. Weston, B. Schölkopf, E. Eskin, C. Leslie, W. Noble, Dealing with large diagonals in kernel matrices, Annals of the Institute of Statistical Mathematics 55 (2003) 391–408.

[57] S. Gudmundsson, T. Runarsson, S. Sigurdsson, Support vector machines and dynamic time warping for time series, in: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, IEEE Computer Society, Washington, DC, USA, 2008, pp. 2772 –2776.

[58] E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei, C. A. Ratanamahatana, The ucr time series classification/clustering data sets homepage (2011).
URL `www.cs.ucr.edu/∼eamonn/time_series_data/`

[59] A. Frank, A. Asuncion, The uci machine learning repository (2011).
URL `http://archive.ics.uci.edu/ml`

[60] Y. Zheng, L. Zhang, X. Xie, W.-Y. Ma, Mining interesting locations and travel sequences from gps trajectories, in: Proceedings of 18th International Conference on World Wide Web, WWW'09, ACM, New York, NY, USA, 2009, pp. 791–800.

[61] S. Salzberg, On comparing classifiers: pitfalls to avoid and a recommended approach, Data Mining and Knowledge Discovery 1 (3) (1997) 317–327.

[62] P. Langley, Machine learning as an experimental science, Machine Learning 3 (1) (1988) 5–8.

[63] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (3) (2011) 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/∼cjlin/libsvm`.

[64] S. Salvador, P. Chan, Toward accurate dynamic time warping in linear time and space, Intell. Data Anal. 11 (5) (2007) 561–580.

[65] A. Bordes, S. Ertekin, J. Weston, L. Bottou, Fast kernel classifiers with online and activate learning, Journal of Machine Learning Research 6 (2005) 1579–1619.

[66] A. B. A. Graf, A. J. Smola, S. Borer, Classification in a normalized feature space using support vector machines, IEEE Transactions on Neural Networks 14 (3) (2003) 597–605.

[67] S. Ali, K. Smith-Miles, Improved support vector machine generalization using normalized input space, in: A. Sattar, B.-h. Kang (Eds.), AI 2006: Advances in Artificial Intelligence, Vol. 4304 of Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg, 2006, pp. 362–371.

[68] C. A. Ratanamahatana, E. Keogh, Making time-series classification more accurate using learned constraints, in: M. W. Berry, U. Dayal, C. Kamath, D. Skillicorn (Eds.), Proceedings of the Fourth SIAM International Conference on Data Mining, SIAM, Philadelphia, PA 19104-2688 USA, 2004, pp. 11–22.

[69] T. Fawcett, An introduction to roc analysis, Pattern Recognition Letters 27 (2006) 861–874.

[70] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, K. Obermayer, Classification on pairwise proximity data, in: S. A. Solla, T. K. Leen, K.-R. Müller (Eds.), Advances in Neural Information Processing Systems, Vol. 11, The MIT Press, 1999, pp. 438–444.

[71] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (200) (1937) 675–701.

[72] O. J. Dunn, Multiple comparisons among means, Journal of the American Statistical Association 56 (293) (1961) pp. 52–64.

[73] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[74] A. Fisher, Statistical methods and scentific inference, 2nd Edition, Hafner Publishing Co., New York, USA, 1959.

[75] Y. Zheng, X. Xie, W.-Y. Ma, Geolife: A collaborative social networking service among user, location and trajectory, IEEE Database Engineering Bulletin 33 (2) (2010) 32–39.

[76] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, New York, NY, USA, 2004.

[77] L. v. d. Maaten, Learning discriminative fisher kernels, in: Proceedings of 28th International Conference on Machine Learning, Bellevue, WA, USA, 2011, only online version was available at the time of preparing this paper: www.icml-2011.org/papers/178_icmlpaper.pdf.

[78] Y. Zheng, L. Liu, L. Wang, X. Xie, Learning transportation mode from raw gps data for geographic applications on the web, in: Proceedings of the 17th International Conference on World Wide Web, WWW'08, ACM, New York, NY, USA, 2008, pp. 247–256.

[79] M. H. Nguyen, F. D. la Torre, Maximum margin temporal clustering., Journal of Machine Learning Research - Proceedings Track 22 (2012) 520–528.

[80] M. H. Nguyen, Segment-based svms for time series analysis, Ph.D. thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213 (2012).