Learning a Kernel Matrix for Time Series Data from DTW Distances

Hiroyuki Narita, Yasumasa Sawamura, and Akira Hayashi

Graduate School of Information Sciences, Hiroshima City University
3-4-1 Ozuka-Higashi, Asa-Minami-Ku, Hiroshima, 731-3194, Japan
narita@robotics.im.hiroshima-cu.ac.jp

Abstract. One of the advantages of the kernel methods is that they can deal with various kinds of objects, not necessarily vectorial data with a fixed number of attributes. In this paper, we develop kernels for time series data using dynamic time warping (DTW) distances. Since DTW distances are pseudo distances that do not satisfy the triangle inequality, a kernel matrix based on them is not positive semidefinite, in general. We use semidefinite programming (SDP) to guarantee the positive definiteness of a kernel matrix. We present neighborhood preserving embedding (NPE), an SDP formulation to obtain a kernel matrix that best preserves the local geometry of time series data. We also present an out-of-sample extension (OSE) for NPE. We use two applications, time series classification and time series embedding for similarity search to validate our approach.

1 Introduction

We have seen significant development of kernel methods for machine learning in the last decade [1]. Typical kernel method algorithms include support vector machines (SVMs) [2] for large margin classification, and kernel principal component analysis (KPCA) [3] for nonlinear dimensionality reduction. Symmetric positive semidefinite kernel functions that give similarity between objects, play a central role in kernel methods. One of the advantages of these kernel methods is that they can deal with various kinds of objects, not necessarily vectorial data with a fixed number of attributes. Such objects include strings, graphs, and weighted automata.

In this paper, we develop kernels for time series data using dynamic time warping (DTW) distances. Machine learning and data mining on time series data (also known as sequence data), such as speech, gesture, handwriting, and so on, has recently attracted more and more attention from the research community. The DTW distance is a frequently used dissimilarity measure for time series data [4]. Shimodaira et al. [5] proposed a dynamic time alignment kernel for voice recognition, and have reported better classification accuracy than HMMs when the number of training data is small. Bahlmann et al. [6] proposed the GDTW kernel, which substitutes the distance term in a Gaussian kernel with a DTW distance, and which achieves classification accuracy comparable with

M. Ishikawa et al. (Eds.): ICONIP 2007, Part II, LNCS 4985, pp. 336–345, 2008.

that of HMMs for online handwritten characters. However, since DTW distances are pseudo distances that do not satisfy the triangle inequality, the previous approaches have failed to prove the positive semidefiniteness of the kernel matrix.

In order to guarantee the positive semidefiniteness of a kernel matrix, we use semidefinite programming (SDP) [7]. SDP has been used in machine learning to optimize a kernel matrix [8] for classification, and also to find low dimensional manifolds [9,10]. We present neighborhood preserving embedding (NPE), an SDP formulation, to obtain a kernel matrix that best preserves the local geometry of time series data in terms of the DTW distances. We also present an out-of-sample extension (OSE) for NPE.

We use two applications, time series classification [11] and time series embedding for similarity search [12], to validate our approach. In time series classification, the well known kernel trick is used to map time series data into a high dimensional feature space for linear separability and larger margin. On the other hand, in time series embedding for similarity search, a low dimensional feature space is sought for efficient multidimensional search. We present a suitable SDP formulation for the purpose.

The rest of this paper is organized as follows. In Section 2, we review DTW distances. In Section 3, we explain how to construct a kernel matrix from DTW distances using SDP. The resulting kernel matrix is used for large margin classification in Section 4. It is also used for low dimensional embedding via kernel PCA in Section 5. We conclude in Section 6.

2 Dynamic Time Warping(DTW)

A set of n time series data, $\mathcal{X} = \{X_1, \ldots, X_n\}$, is given, where X_i $(1 \leq i \leq n)$ is a sequence of feature vectors whose length is l_i $X_i = (\boldsymbol{x}^i_1, \ldots, \boldsymbol{x}^i_{l_i})$. DTW finds the smallest distance, i.e., the maximal similarity, between the time series data through all nonlinear time warping that corresponds to a change in time scale [4]. In this paper, we use the DTW distances that are computed as follows, where $\| \cdot \|$ is the Euclidean norm.

1. Initial and boundary conditions.
   - start : $g(1,1) = 0$
   - end$g(l_i, l_j)$
   - boundary $g(t_i, 0) = g(0, t_j) = \infty$
2. Repeat
   for $1 \leq t_i \leq l_i 1 \leq t_j \leq l_j$

$$g(t_i, t_j) = \min \begin{cases} g(t_i - 1, t_j) + \|\boldsymbol{x}^i_{t_i} - \boldsymbol{x}^j_{t_j}\|^2 \\ g(t_i - 1, t_j - 1) + 2\|\boldsymbol{x}^i_{t_i} - \boldsymbol{x}^j_{t_j}\|^2 \\ g(t_i, t_j - 1) + \|\boldsymbol{x}^i_{t_i} - \boldsymbol{x}^j_{t_j}\|^2 \end{cases} \tag{1}$$

3. Finish $d^2(X_i, X_j) = g(l_i, l_j)$

3 Learning a Kernel Matrix from DTW Distances

Let $\boldsymbol{\Phi}$ be a mapping from time series data into a feature space $\boldsymbol{\mathcal{F}}$.

$$\begin{aligned} \text{mapping } \boldsymbol{\Phi} : \mathcal{X} &\to \boldsymbol{\mathcal{F}} \\ X_i &\mapsto \boldsymbol{\Phi}(X_i) \end{aligned}$$

In what follows, we write $\boldsymbol{K} \succeq 0$ as an abbreviation for $\boldsymbol{K}$ being a symmetric matrix that satisfies positive semidefiniteness. Our approach is to learn a kernel matrix $\boldsymbol{K} \succeq 0$, $\boldsymbol{K}(i,j) = \langle \boldsymbol{\Phi}(X_i), \boldsymbol{\Phi}(X_j) \rangle (1 \leq i, j \leq n)$ from DTW distances using the following well known relationship between distances and inner products.

$$\begin{aligned} d^2(X_i, X_j) = ||\boldsymbol{\Phi}(X_i) - \boldsymbol{\Phi}(X_j)||^2 &= \langle \boldsymbol{\Phi}(X_i) - \boldsymbol{\Phi}(X_j), \boldsymbol{\Phi}(X_i) - \boldsymbol{\Phi}(X_j) \rangle \\ &= \boldsymbol{K}(i,i) - \boldsymbol{K}(i,j) - \boldsymbol{K}(j,i) + \boldsymbol{K}(j,j) \end{aligned}$$

3.1 Neighborhood Preserving Embedding (NPE)

DTW distances are pattern matching scores, so it is known that smaller distances are reliable, but larger distances are unreliable [11]. Therefore, it is expected that a mapping that pays attention only to neighborhood distances will have better results. Here we introduce Neighborhood Preserving Embedding (NPE), that learns a kernel matrix $\boldsymbol{K} \succeq 0$ that best preserves squared neighborhood distances. NPE entails the following procedure :

1. For a given n time series data $\{X_1, \ldots, X_n\}$, compute the DTW distance $\{d(X_i, X_j) | 1 \leq i, j \leq n\}$ between all data pairs.
2. Solve the following optimization problem by SDP [7].

$$\min_{\boldsymbol{K} \succeq 0} \sum_{i=1}^{n} \sum_{j : X_j \sim X_i} w_{ij} |d^2(X_i, X_j) - \langle \boldsymbol{B}_{ij}, \boldsymbol{K} \rangle| \tag{2}$$

$$s.t. \sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{K}(i,j) = 0,$$

where "$X_j \sim X_i$" denotes that X_j is a neighbor of X_i and w_{ij} is a weight parameter. $\boldsymbol{B}_{ij}$ is a sparse $n \times n$ matrix used to compute square distances from $\boldsymbol{K}$, that is $\boldsymbol{B}_{ij}(i,i) = \boldsymbol{B}_{ij}(j,j) = 1, \boldsymbol{B}_{ij}(i,j) = \boldsymbol{B}_{ij}(j,i) = -1$ and all other elements are 0. Note that "$\langle \cdot, \cdot \rangle$" in Eq.(2) is an inner product operator between matrices.

$\sum_i \sum_j \boldsymbol{K}(i,j) = 0$ is the well known constraint for *centering* $\boldsymbol{K}$. Since $\sum_i \sum_j \boldsymbol{K}(i,j) = 0 \Leftrightarrow \| \sum_i \boldsymbol{\Phi}(X_i) \|^2 = 0 \Leftrightarrow \sum_i \boldsymbol{\Phi}(X_i) = \mathbf{0}$ holds, the constraint causes the center of gravity of the feature vectors $\{\boldsymbol{\Phi}(X_i) | 1 \leq i \leq n\}$ to move to the origin. This is required in order to apply kernel PCA later for dimensionality reduction.

3. We eigen-decompose the kernel matrix $\boldsymbol{K}$, that is optimized in step 2 above. The decomposed matrix is expressed as follows.
$$\boldsymbol{K} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T, \tag{3}$$
where $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \lambda_1 \geq \ldots \geq \lambda_n \geq 0$ is a diagonal matrix of the eigenvalues, and $\boldsymbol{U} = [\boldsymbol{e}^1, \ldots, \boldsymbol{e}^n]$ is a matrix of the eigenvectors.
Let us denote $\boldsymbol{\Phi}(X_i)$ as $\boldsymbol{\Phi}_i$. Since $\boldsymbol{K} = [\boldsymbol{\Phi}_1\boldsymbol{\Phi}_2, \ldots \boldsymbol{\Phi}_n]^T[\boldsymbol{\Phi}_1\boldsymbol{\Phi}_2, \ldots \boldsymbol{\Phi}_n]$ holds, Eq.(3) gives
$$[\boldsymbol{\Phi}_1\boldsymbol{\Phi}_2, \ldots \boldsymbol{\Phi}_n] = \boldsymbol{\Lambda}^{1/2}\boldsymbol{U}^T \tag{4}$$
$$\boldsymbol{\Phi}_i(k) = \sqrt{\lambda_k}\boldsymbol{e}_k(i) \qquad (1 \leq k \leq p) \qquad \forall i \tag{5}$$
where $\boldsymbol{\Phi}_i(k)$ is the kth entry of $\boldsymbol{\Phi}_i$, $e_k(i)$ is the ith entry of the kth eigenvector $\boldsymbol{e}_k$, and p is the rank of $\boldsymbol{K}$.

As for the neighborhood relationship, we have two choices. We define the ϵ-neighborhood relationship as $X_i \sim X_j \Leftrightarrow d(X_i, X_j) < \epsilon$. The symmetric k-nn neighborhood relationship is defined as $X_i \sim X_j \Leftrightarrow X_i \in knn(X_j) \vee X_j \in knn(X_i)$, where $knn(X_i)$ is the set of k nearest neighbors of X_i.

3.2 Out-of-Sample Extension (OSE)

Given additional time series data, X_{n+1}, it is natural to use NPE again to obtain an $(n+1)\times(n+1)$ kernel matrix $\boldsymbol{K}_{n+1}$. However, this adds a heavy computational load. We therefore introduce Out-of-Sample Extension (OSE) to obtain a suboptimal kernel matrix $\tilde{\boldsymbol{K}}_{n+1}$ by expanding the kernel matrix $\boldsymbol{K}_n$ that has already been computed by NPE. We define an extended kernel matrix $\tilde{\boldsymbol{K}}_{n+1}$ as follows:
$$\tilde{\boldsymbol{K}}_{n+1} = \begin{bmatrix} \boldsymbol{K}_n & \boldsymbol{b} \\ \boldsymbol{b}^T & c \end{bmatrix} \succeq 0, \tag{6}$$
$$\boldsymbol{b} = (\langle \boldsymbol{\Phi}_1, \boldsymbol{\Phi}_{n+1} \rangle, \langle \boldsymbol{\Phi}_2, \boldsymbol{\Phi}_{n+1} \rangle, \ldots, \langle \boldsymbol{\Phi}_n, \boldsymbol{\Phi}_{n+1} \rangle)^T \tag{7}$$
$$c = \langle \boldsymbol{\Phi}_{n+1}, \boldsymbol{\Phi}_{n+1} \rangle \tag{8}$$

Then, $\tilde{\boldsymbol{K}}_{n+1}, \boldsymbol{b} \in \boldsymbol{R}^n$, and $c \in \boldsymbol{R}$ are obtained by solving the following SDP.
$$\min_{\tilde{\boldsymbol{K}}_{n+1} \succeq 0, \boldsymbol{b}, c} \sum_{i: X_i \sim X_{n+1}} w_{i,n+1} | d^2(X_i, X_{n+1}) - \langle \boldsymbol{B}_{i,n+1}, \tilde{\boldsymbol{K}}_{n+1} \rangle | \tag{9}$$
$$s.t. \quad \tilde{\boldsymbol{K}}_{n+1} = \begin{bmatrix} \boldsymbol{K}_n & \boldsymbol{b} \\ \boldsymbol{b}^T & c \end{bmatrix}$$

Finally, we consider embedding the additional time series data, X_{n+1}, into the space in which $\{X_i | 1 \leq i \leq n\}$ are already embedded using Eq.(5). Let $\tilde{\boldsymbol{\Phi}}_{n+1}$ be the projection of $\boldsymbol{\Phi}_{n+1}$ into the space spanned by $\{\boldsymbol{\Phi}_i | 1 \leq i \leq n\}$. Substituting Eq.(4) into Eq.(7) yields $(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2})\tilde{\boldsymbol{\Phi}}_{n+1} = \boldsymbol{b}$. Hence, we obtain the following.
$$\tilde{\boldsymbol{\Phi}}_{n+1} = (\boldsymbol{U}\boldsymbol{\Lambda}^{1/2})^{\dagger}\boldsymbol{b} \tag{10}$$
$$\tilde{\boldsymbol{\Phi}}_{n+1}(k) = \frac{1}{\sqrt{\lambda_k}}\boldsymbol{e}_k^T\boldsymbol{b}, \qquad (1 \leq k \leq p) \tag{11}$$
where $(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2})^{\dagger}$ is the pseudo inverse of $(\boldsymbol{U}\boldsymbol{\Lambda}^{1/2})$ and p is the rank of $\boldsymbol{K}_n$.

4 Large Margin Classification

In this section, we classify time series data by SVM. We employ linear, polynomial, and RBF kernels.

$$\begin{aligned}\text{Linear kernel}: \ & \boldsymbol{K}^{lin}(i,j) = \langle \boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j \rangle \\ \text{Polynomial kernel}: \ & \boldsymbol{K}^{pol}(i,j) = (1 + \langle \boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j \rangle)^p \\ \text{RBF kernel}: \ & \boldsymbol{K}^{rbf}(i,j) = exp(-\|\boldsymbol{\Phi}_i - \boldsymbol{\Phi}_j\|^2/2\gamma^2),\end{aligned}$$

where $\boldsymbol{\Phi}_i$ $(1 \leq i \leq n+1)$ is the feature vector for X_i obtained by NPE and OSE using Eqs.(5) and (11) [1], and γ is the parameter for the RBF kernel. Note that since the linear kernels are positive semidefinite, the polynomial and RBF kernels are also positive semidefinite.

4.1 UNIPEN

The UNIPEN-DTW data[13] consists of DTW distance matrices that are based on the UNIPEN Train-R01/V07 online handwriting sequence dataset. The data contains 2 sets with 250 samples per set from 5 classes ('a' to 'e').

We conducted the multi class classification experiment in two settings.

- Transductive setting. (1) Both the training data and the test data are embedded by NPE. (2) The classifier is trained with the training data, and the test data is classified.
- Sequential setting. (1) The training data is embedded by NPE, and the classifier is trained. (2) Then, the test data, embedded by OSE, is classified.

To solve the SDP optimization problems in NPE and OSE, we use publicly available software SDPT3 [14]. We set the parameter $w_{ij} = 1$ for all i, j pairs and use a k-nn neighborhood, $k = 6$, for both NPE and OSE. Since the data has turned out to be linearly separable[2], we tested only hard margin SVMs, adjusting p for $\boldsymbol{K}^{pol}$ and γ for $\boldsymbol{K}^{rbf}$. We use *one-versus-the-rest* SVM as multiclass SVM.

We compare our results with those for the following *distance substitution*(DS) kernels [13].

$$\begin{aligned}\text{Linear distance kernel}: \ & \boldsymbol{K}_d^{lin}(i,j) = \langle X_i, X_j \rangle_d \\ \text{Polynomial distance kernel}: \ & \boldsymbol{K}_d^{pol}(i,j) = (1 + \gamma\langle X_i, X_j \rangle_d)^p \\ \text{RBF distance kernel}: \ & \boldsymbol{K}_d^{rbf}(i,j) = exp(-d^2(X_i, X_j)/2\gamma^2),\end{aligned}$$

where $\langle X_i, X_j \rangle_d = -1/2(d^2(X_i, X_j) - d^2(X_i, O) - d^2(X_j, O))$. O is the origin and was chosen as the point with the minimum squared distance sum relative to the other training data. Since DTW distances are pseudo distances, the distance

[1] In this section, we omit the tilde on top of $\tilde{\boldsymbol{\Phi}}_{n+1}$ to simplify the notation.

[2] Assuming $\boldsymbol{K}^{lin}$ is of full rank, its feature space dimension is n, the number of the training data. Hence, the VC dimension for $\boldsymbol{K}^{lin}$ is n+1.

Table 1. LOO-errors for UNIPEN. The error rates for NPD, CSE, RNE, 1-nn, and k-nn are from [13]. As for the k-nn classifier, the best k-nn are shown. Tra and Seq refer to the transductive and sequential settings, respectively. The order of $\boldsymbol{K}^{pol}$ is 3 for both datasets. The value of γ for $\boldsymbol{K}_d^{rbf}$ is 1.0 except for Tra in dataset #2, where it is 0.75.

dataset	$\boldsymbol{K}_d^{pol}$			$\boldsymbol{K}_d^{rbf}$			1-nn	k-nn	$\boldsymbol{K}^{lin}$		$\boldsymbol{K}^{pol}$		$\boldsymbol{K}^{rbf}$	
	NPD	CNE	RNE	NPD	CNE	RNE			Tra	Seq	Tra	Seq	Tra	Seq
#1	6.0	5.2	5.6	5.2	4.4	4.8	5.6	5.6	5.2	6.0	4.0	4.8	4.0	5.2
#2	7.6	6.8	6.4	6.0	6.0	5.6	7.2	6.4	6.8	6.0	6.0	4.8	6.4	5.2

substitution kernels are Not Positive semiDefinite (NPD) kernels. To transform NPD kernels to be positive semidefinite, two methods are provided. Cutting off Negative Eingenvalues (CNE) cuts off contributions corresponding to negative eigenvalues. Reflecting Negative Eingenvalues (RNE) reflects the negative eigenvalues by taking their absolute values. Note that CNE and RNE can be used only under the transductive setting.

The result is evaluated by leave-one-out (LOO) errors. See Table 1. In the transductive setting (Tra), our polynomial and RBF kernels, $\boldsymbol{K}^{pol}$ and $\boldsymbol{K}^{rbf}$, respectively, generally perform better for both datasets than CNE and RNE of the corresponding DS-kernels, $\boldsymbol{K}_d^{pol}$ and $\boldsymbol{K}_d^{rbf}$, respectively. The exception is that our rbf kernel has a larger error rate for the second dataset. In the sequential setting (Seq), our kernels always perform better than the corresponding NPD kernels. In addition, our kernels also perform better than 1-nn and k-nn classifiers. We are currently working hard to investigate the reason why all of our kernels perform better in the sequential setting (i.e., using NPE + OSE) than in the transductive setting (i.e., using only NPE) for the second dataset.

Table 2 shows how the size of k-nn neighborhoods influences the SVM classifications. Due to the relaiablility of smaller DTW distance, relatively small k values bring better results.

5 Low Dimensional Embedding for Similarity Search

In this section, we consider how to speed up a similarity search of time series data, when dissimilarity is defined in terms of DTW distances. Stated more

Table 2. LOO-errors for UNIPEM with k-nn neighborhoods ($6 \leq k \leq 250$). All errors are computed by linear SVM with NPE.

	$\boldsymbol{K}^{lin}$								
dataset	k = 6	k = 8	k=12	k=15	k = 20	k = 50	k = 80	k = 150	k = 250
#1	5.2	5.6	4.8	6.8	14.8	14.8	11.6	19.6	16.0
#2	6.8	6.4	10.0	6.4	13.6	16.8	10.4	12.8	-

concretely, we consider the following problem. A set of n time series data (time series DB): $\mathcal{X} = \{X_1, \ldots, X_n\}$, is given. Given a query Q, another time series data, *quickly* find the k nearest neighbors of Q, i.e., find the k $X_i's$ with the smallest DTW distances.

5.1 Proposed Method

We adopt the approach of embedding time series data in a low dimensional Euclidean space with KPCA[3], and performing a multidimensional search. The time complexity of nearest neighbor search in the embedded space using the kd-tree is $O(\log n)$ [15], whereas that of the linear search is $O(n)$, where n is the number of data. In order to speed up the similarity search, the key issue is how to embed the data accurately (1) into a *low* dimensional space (2) from a *small* number of DTW distances.

Lower dimensional embedding is preferred because the complexity of the kd-tree search increases exponentially as the number of embedding dimensions p grows. For our purposes, we introduce NPE with regularization by adding a regularization term to the objective function in Eq. (2):

$$\min_{\boldsymbol{K} \succeq 0} \sum_i \sum_{j \in \boldsymbol{N}_i} w_{ij} |d^2(X_i, X_j) - \langle \boldsymbol{B}_{ij}, \boldsymbol{K} \rangle| + \eta \cdot \mathrm{tr}(\boldsymbol{K}), \qquad (12)$$

where $\mathrm{tr}(\boldsymbol{K})$ is the trace of $\boldsymbol{K}$ and η is a parameter to trade off the two terms in the objective function. It can be shown that $\mathrm{tr}(\boldsymbol{K}) = 1/(2n) \sum_i \sum_j ||\boldsymbol{\Phi}_i - \boldsymbol{\Phi}_j||^2$, i.e. $\mathrm{tr}(\boldsymbol{K})$ is proportional to the variance of data in the feature space. We promote low dimensional embedding by adjusting η.

To embed the data from a small number of DTW distances, we use OSE. We randomly select m ($m \ll n$) samples from n time series data in the DB, and apply NPE to m samples. The remaining non samples and the query are embedded by OSE using DTW distances to the m samples.

5.2 Experiment

The objective of this experiment is to evaluate the accuracy of low dimensional embedding using NPE and OSE. For two kinds of time series data (ASL [3] and ISOLET [4]), we compare our method with multidimensional scaling (MDS) [18]. We use the Nyström method [19] as an out-of-sample extension for MDS.

We adjust η in Eq. (2) so as to embed the data in a low dimensional space. Fig. 1 shows the eigenvalue distribution for ASL when η is changed.

For the task, we choose to search for 10 nearest neighbors (NNs) in the time series DB. We compute recall-precision (RP) curves for each embedding method.

[3] ASL is based on Australian sign Language data in the UCI KDD Archive [16]. The data consist of 95 signed words.

[4] ISOLET is a database of letters of the English alphabet spoken in isolation [17]. The database consists of 7800 spoken letters, two productions of each letter by 150 speakers.

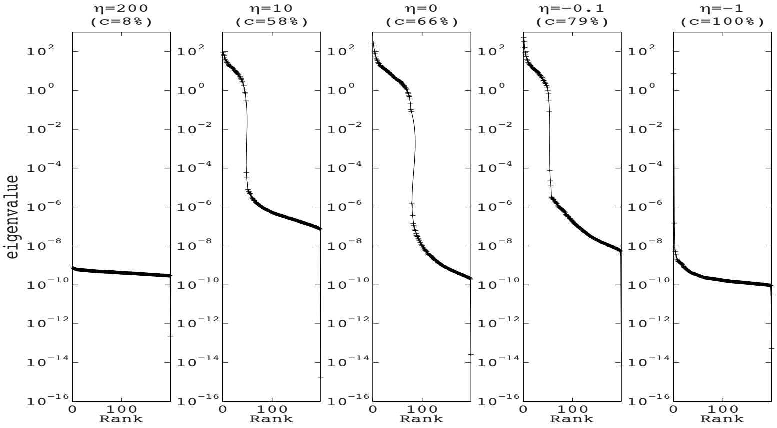

Fig. 1. The eigenvalue distribution of the kernel matrix for the ASL sample data. The *contribution rate* c under the embedding dimension p, $c = \sum_{i=1}^{p} \lambda_i / \sum_{j=1}^{m} \mathrm{tr}(\boldsymbol{K})$ is also shown. As η decreases, big eigenvalues become dominant. Although, the rightmost image shows the highest contribution rate, the number of nonzero eigenvalues is only one, therefore the accuracy that preserves distances has been lost.

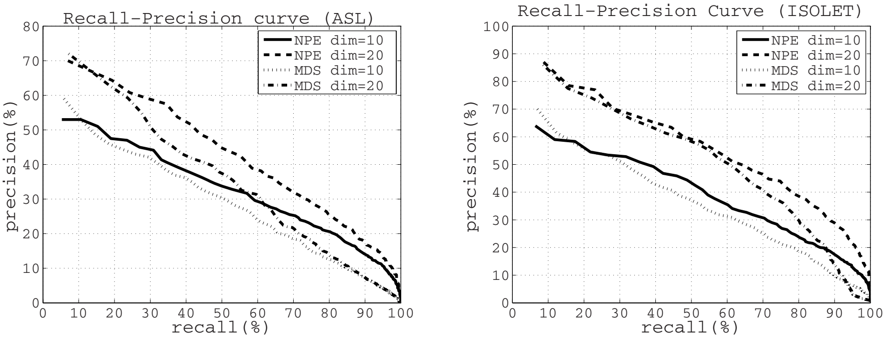

Fig. 2. RP curves for NPE and MDS. We set $w_{ij} = 1$ for all i,j pairs in Eqs. (12) and (9), used an ϵ neighborhood. The value of ϵ was selected so that each datum has at least 20 neighbors from the samples. DB size, $n = 3000$, and sample size, $m = 200$, the embedding dimension, $p = 10$,20. The average of 100 queries was taken. **(left)** ASL: We use as DB time series examples for 43 words, such as "change","deaf","glad","her", and "innocent", which have similar words. We use examples for "lose" and "love" as query time series. **(right)** ISOLET: We randomly selected data from the dataset and used thse as DB and as queries. The 28-dimensional feature vector consists of 14 MFCCs and their first-order time derivatives.

We view up to k ($k > 10$) NNs in the *embedded* space as retrieved (*positive*) results, and count how many of them are *true*, i.e., are within 10 NNs in terms of DTW distance.

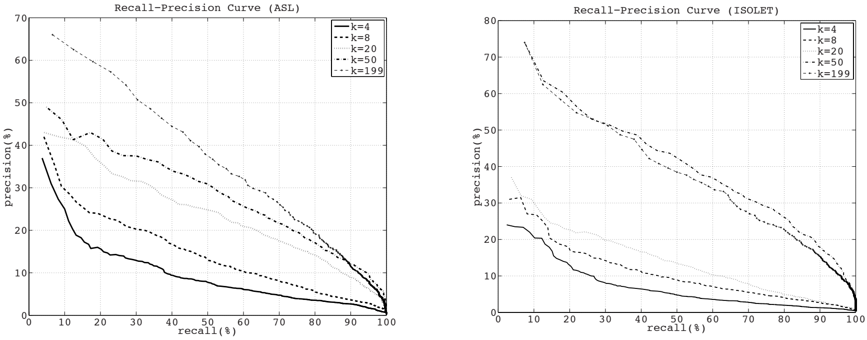

Fig. 3. RP Curves for NPE for k-nn neighborhoods where $k = 4, 8, 20, 50, 199$. $n = 3000$, $m = 200$, $p = 10$. The average of 100 queries. **(left)** ASL. **(right)** ISOLET.

Fig. 2 shows the RP curves for the ASL and ISOLET data. We see from the figure that NPE performs better than MDS. We attribute the reason to the fact that NPE constructs the kernel using only neighborhood distances, and it has no negative eigenvalues.

To examine the effect of the neighborhood size, we also experimented using k-nn neighborhoods for various k values. Fig. 3 shows the RP curves for the ASL and ISOLET. [5]

6 Conclusion

We have developed kernels for time series data from DTW distances. By using SDP, we can guarantee the positive definiteness of the kernel matrix. We have presented NPE, an SDP formulation to obtain a kernel matrix that best preserves the local geometry of time series data, together with its out-of-sample extension. We have shown two applications, time series classification and time series embedding for similarity search in order to validate our approach.

References

1. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
2. Corres, C., Vapnik, V.: Support vector networks. Machine Learning 20, 273–297 (1995)
3. Schölkopf, B., Smola, A., Müller, K.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10, 1299–1319 (1998)
4. Rabiner, L., Juang, B.: Fundamentals of Speech Recognition. Prentice-Hall, Englewood Cliffs (1993)

[5] Contrary to our expectation, larger neighborhood size generally leads to better results. It seems that low dimensional embedding is difficult with small neighborhood.

5. Shimodaira, H., Noma, K., Nakai, M., Sagayama, S.: Dynamic time-alignment kernel in support vector machine. In: Neural Information Processing Systems 14, pp. 921–928. MIT Press, Cambridge (2002)
6. Bahlmann, C., Haasdonk, B., Burkhardt, H.: On-line handwriting recognition with support vector machines-a kernel approach. In: Proc. 8th Int. W/S on Frontiers in Handwriting Recognition, pp. 49–54 (2002)
7. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Rev. 38(1), 49–95 (1996)
8. Lanckriet, G., Christianini, N., Barlett, P., Ghaoui, L., Jordan, M.: Learning the kernel matrix with semidifinite programming. Journal of Machine Learning Research 5, 27–72 (2004)
9. Weinberger, K.Q., Sha, F., Saul, L.K.: Learning a kernel matrix for nonlinear dimensionality reduction. In: Proc. 21st Int. Conf. on Machine Learning (ICML 2004), pp. 839–846 (2004)
10. Lu, F., Keles, S., Wright, S., Wahba, G.: Framework for kernel regularization with application to protein clustering. PNAS 102(35), 12332–12337 (2005)
11. Hayashi, A., Mizuhara, Y., Suematsu, N.: Embedding time series data for classification. In: Perner, P., Imiya, A. (eds.) MLDM 2005. LNCS (LNAI), vol. 3587, pp. 356–365. Springer, Heidelberg (2005)
12. Hayashi, A., Nisizaki, K., Suematsu, N.: Fast similarity search of time series data using the nystrom method. In: ICDM 2005 Workshop on Temporal Data Mining, pp. 157–164 (2005)
13. Haasdonk, B., Bahlmann, C.: Learning with distance substitution kernels. In: Rasmussen, C.E., Bülthoff, H.H., Schölkopf, B., Giese, M.A. (eds.) DAGM 2004. LNCS, vol. 3175, pp. 220–227. Springer, Heidelberg (2004)
14. Toh, K., Tütüncü, R., Todd, M.: Solving semidefinite-quadratic-linear programming using sdpt3. Mathematical Programming 95, 180–217 (2003)
15. Friedman, J., Bentley, J., Finkel, R.: An algorithm for finding the best matches in logarithmic expected time. ACM Trans. Mathematical Software 3(3), 209–226 (1977)
16. Kadous, W.: Australian sign language data in the uci kdd archive (1995), `http://www.cse.unsw.edu.au/~waleed/tml/data/`
17. Cole, R., Muthusamy, Y., Fanty, M.: The ISOLET spoken letter database. Technical Report CS/E 90-004 (1990)
18. Cox, T., Cox, M.: Multidimensional Scaling. Chapman and Hall, Boca Raton (2001)
19. Bengio, Y., Vincent, P., Paiement, J.: Learning eigenfunctions links spectral embedding and kernel pca. Neural Computation 16(10), 2197–2219 (2004)