DTW-Distance Based Kernel for Time Series Data

Hiroyuki NARITA(\textsuperscript{a}), Member, Yasumasa SAWAMURA(\textsuperscript{b}), Nonmember, and Akira HAYASHI(\textsuperscript{a}), Member

SUMMARY 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.

key words: DTW, SDP, SVM

1. Introduction

We have seen significant development of kernel methods for machine learning in the last decade\cite{1}. Typical kernel method algorithms include support vector machines (SVMs)\cite{2} for large margin classification and kernel principal component analysis (KPCA)\cite{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, have recently attracted more and more attention from the research community. The DTW distance is a frequently used dissimilarity measure for time series data\cite{4}. Shimodaira et al.\cite{5} proposed a dynamic time alignment kernel for voice recognition, and have reported better classification accuracy comparable with that of HMMs for online handwriting. Bahliamn et al.\cite{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 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.

To guarantee the positive semidefiniteness of a kernel matrix, we use semidefinite programming (SDP)\cite{7}. An SDP formulation called neighborhood preserving embedding (NPE) is introduced to obtain a kernel matrix that best preserves the local geometry of the time series data in terms of the DTW distances. We also present an out-of-sample extension (OSE) for NPE. SDP has been used in machine learning both to optimize a kernel matrix\cite{8} for classification, and also to find low-dimensional manifolds from distances\cite{9,10}. The problem of learning a kernel matrix from distances is not, however, addressed in\cite{8}. Moreover, the formulation in\cite{9} requires that the distances used for embedding satisfy the triangle inequality. Consequently, this formulation cannot be applied to DTW pseudo distances, which do not satisfy the triangle inequality. Our SDP formulation on the other hand, allows the use of pseudo distances to obtain a kernel matrix. Whereas Lu et al.\cite{10} choose distances randomly to reduce computational load, we only use reliable neighborhood distances for computing the more appropriate kernel matrix. Although the formulations in both\cite{9} and\cite{10} are concerned with low dimensional embedding, this is not of great concern to us for large margin classification.

We use two applications, time series classification\cite{11} and time series embedding for similarity search\cite{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 Sect. 2, we review DTW distances. In Sect. 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 Sect. 4. It is also used for low dimensional embedding via kernel PCA in Sect. 5. We conclude in Sect. 6.

2. Dynamic Time Warping (DTW)

A set of \(n\) time series data, \(X = \{X_1, \ldots, X_n\}\), is given, where \(X_i (1 \leq i \leq n)\) is a sequence of feature vectors whose length is \(l_i; X_i = (x_{i1}, \ldots, x_{il_i})\). 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 discussed in [13] that are computed as follows, where ||·|| is the Euclidean norm.

1. Boundary conditions.
   - \( g(0,0) = 0 \)
   - \( g(t_i,0) = \infty \quad (1 \leq t_i \leq l_i) \)
   - \( g(0,t_j) = \infty \quad (1 \leq t_j \leq l_j) \)

2. Repeat
   for \( 1 \leq t_i \leq l_i, 1 \leq t_j \leq l_j \)
   
   \[
   g(t_i,t_j) = ||x_i^t - x_j^t||^2 + \min \left\{ \frac{g(t_i - 1,t_j)}{g(t_i - 1,t_j - 1)} \right\} (1) 
   \]

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

3. Learning a Kernel Matrix from DTW Distances

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

\[
\Phi : X \rightarrow \mathcal{F} \\
X_i \mapsto \Phi(X_i)
\]

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

\[
d^2(X_i,X_j) = ||\Phi(X_i) - \Phi(X_j)||^2 \\
= \langle \Phi(X_i) - \Phi(X_j), \Phi(X_i) - \Phi(X_j) \rangle \\
= K(i,i) - K(i,j) - K(j,i) + K(j,j)
\]

### 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 \( K \geq 0 \) 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 \( d(X_i,X_j) \) between all data pairs.
2. Solve the following optimization problem by SDP [7].

\[
\min_{K \succeq 0} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d^2(X_i,X_j) - \langle B_{ij}, K \rangle \quad (2)
\]

\[
s.t. \quad \sum_{i=1}^{n} \sum_{j=1}^{n} K(i,j) = 0,
\]

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

\[
\sum_{i} \sum_{j} K(i,j) = 0 \quad \text{is the constraint for centering } K. 
\]

Since \( \sum_{i} \sum_{j} K(i,j) = 0 \Leftrightarrow ||\sum_{i} \Phi(X_i)||^2 = 0 \Leftrightarrow \sum_{i} \Phi(X_i) = 0 \), holds, the constraint causes the center of gravity of the feature vectors \( \{\Phi(X_i)\}_{1 \leq i \leq n} \) 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 \( K \), that is optimized in step 2 above. The decomposed matrix is expressed as follows.

\[
K = U \Lambda U^T, 
\]

where \( \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n), \lambda_1 \geq \ldots \geq \lambda_n \geq 0 \) is a diagonal matrix of the eigenvalues, and \( U = \{e_1, \ldots, e_n\} \) is a matrix of the eigenvectors.

Let us denote \( \Phi(X_i) \) as \( \Phi_i \). Since \( K = [\Phi_1, \Phi_2, \ldots, \Phi_n]^T [\Phi_1, \Phi_2, \ldots, \Phi_n] \) holds, Eq. (3) gives

\[
[\Phi_1, \Phi_2, \ldots, \Phi_n] = \Lambda^{1/2} U^T 
\]

\[
\Phi_i(k) = \sqrt{\lambda_i} e_i(i) \quad (1 \leq k \leq p) \quad \forall i 
\]

where \( \Phi_i(k) \) is the \( k \)th entry of \( \Phi_i \), \( e_i(i) \) is the \( i \)th entry of the \( k \)th eigenvector \( e_k \), and \( p \) is the rank of \( K \).

As for the neighborhood relationship, we have two choices. We define the \( \epsilon \)-neighborhood relationship as \( X_i \sim X_j \Leftrightarrow d(X_i,X_j) < \epsilon \). The symmetric \( k \)-nn neighborhood relationship is defined as \( X_i \sim X_j \Leftrightarrow X_i \in \text{knn}(X_j) \lor X_j \in \text{knn}(X_i) \), where \( \text{knn}(X_i) \) 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 \( K_{n+1} \). However, this adds a heavy computational load. We therefore introduce Out-of-Sample Extension (OSE) to obtain a suboptimal kernel matrix \( \tilde{K}_{n+1} \) by expanding the kernel matrix \( K_n \) that has already been computed by NPE. We define an extended kernel matrix \( \tilde{K}_{n+1} \) as follows:

\[
\tilde{K}_{n+1} = \begin{bmatrix} K_n & b \\ b^T & c \end{bmatrix} \succeq 0, 
\]

\[
b = \langle \Phi_1, \Phi_{n+1} \rangle, \langle \Phi_2, \Phi_{n+1} \rangle, \ldots, \langle \Phi_n, \Phi_{n+1} \rangle \rangle^T 
\]

\[
c = \langle \Phi_{n+1}, \Phi_{n+1} \rangle 
\]

Then, \( \tilde{K}_{n+1}, b \in \mathbb{R}^n \), and \( c \in \mathbb{R} \) are obtained by solving the following SDP.

\[
\min_{K_{n+1} \succeq 0, b,c} \sum_{i,X_n \neq X_{n+1}} w_{i,X_{n}} d^2(X_i,X_{n+1}) - \langle B_{i,X_{n+1}}, \tilde{K}_{n+1} \rangle \quad (7)
\]
\[
\begin{align*}
\text{s.t. } \tilde{K}_{n+1} &= \begin{bmatrix} K_n & b \\ b^T & c \end{bmatrix} \\
\Phi_{n+1} &= (UA^{1/2})^T b \\
\tilde{K}_{n+1}(k) &= \frac{1}{\sqrt{\lambda_k}} e_k^T b, \quad (1 \leq k \leq p)
\end{align*}
\]

where \((UA^{1/2})^T\) is the pseudo inverse of \((UA^{1/2})\) and \(p\) is the rank of \(\tilde{K}_n\).

### 4. Large Margin Classification

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

- **Linear kernel**: \(K_{\text{lin}}(i, j) = \langle \Phi_i, \Phi_j \rangle\).
- **Polynomial kernel**: \(K_{\text{pol}}(i, j) = (1 + \gamma \langle \Phi_i, \Phi_j \rangle)^p\), where \(\Phi, (1 \leq i \leq n+1)\) is the feature vector for \(X_i\) obtained by NPE and OSE using Eqs. (5) and (9), and \(\gamma\) is the parameter for the RBF kernel.
- **RBF kernel**: \(K_{\text{rbf}}(i, j) = \exp(-||\Phi_i - \Phi_j||^2/2\gamma^2)\), where \(\Phi, (1 \leq i \leq n+1)\) is the feature vector for \(X_i\) obtained by NPE and OSE using Eqs. (5) and (9), and \(\gamma\) 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 [14] 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 [15] (see Appendix A). Since the data has turned out to be linearly separable\(^\dagger\), we tested only hard margin SVMs, adjusting \(p\) for \(K_{\text{pol}}\) and \(\gamma\) for \(K_{\text{rbf}}\). We use one-versus-the-rest SVM as multiclass SVM.

We compare our results with those for the following \(distance \ substitution\) (DS) kernels [14].

- **Linear distance kernel**: \(K_{\text{lin}}^{\text{us}}(i, j) = \langle X_i, X_j \rangle_d\).
- **Polynomial distance kernel**: \(K_{\text{pol}}^{\text{us}}(i, j) = (1 + \gamma \langle X_i, X_j \rangle_d)^p\), 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 substitution kernels are Not Positive semiDefinite (NPD) kernels. To transform NPD kernels to be positive semidefinite, two methods are provided. Cutting off Negative Eigenvalues (CNE) cuts off contributions corresponding to negative eigenvalues. Reflecting Negative Eigenvalues (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, \(K_{\text{pol}}^{\text{us}}\) and \(K_{\text{rbf}}^{\text{us}}\), respectively, generally perform better for both datasets than CNE and RNE of the corresponding DS-kernels, \(K_{\text{pol}}^{\text{us}}\) and \(K_{\text{rbf}}^{\text{us}}\). The exception is that our RBF kernel has a higher 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 investigating 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 reliability of smaller DTW distances, relatively small \(k\) values bring better results.

Many papers report that the SDP approach requires a high computational cost. Here we report on the computational cost of our method in computing the LOO error for the UNIPEN-1 data set (250 samples). In a sequential setting (OSE), we only need to extend the kernel matrix that has already been computed, and the time to compute the LOO error is 117.215 seconds in total. This is less than 0.5 seconds for each sample, given the remaining 249 samples as training data. In a transductive setting (NPE), a kernel matrix is learned from scratch every time new test data is obtained, and the time to compute the LOO error is 1993.119 seconds in total.

### 5. Low Dimensional Embedding for Similarity Search

In this section, we consider how to speed up a similarity search.\(^\dagger\)Assuming \(K_{\text{lin}}^{\text{us}}\) is of full rank, its feature space dimension is \(n\), the number of the training data. Hence, the VC dimension for \(K_{\text{lin}}^{\text{us}}\) is \(n+1\).
search of time series data, when dissimilarity is defined in terms of DTW distances. Stated more concretely, we consider the following problem. A set of \( n \) time series data (time series DB): \( X = \{X_1, \ldots, X_n\} \), 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 \( O(\log n) \) [16], 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 \( p \) grows. For our purposes, we introduce NPE with regularization by adding a regularization term to the objective function in Eq. (2):

\[
\begin{align*}
\min_{K \geq 0} & \sum_{i \in N_q} \sum_{j \in N_i} w_{ij} d^2(X_i, X_j) - \left( B_{ij}, K \right) \\
& + \eta \cdot \text{tr}(K),
\end{align*}
\]

where \( \text{tr}(K) \) is the trace of \( K \) and \( \eta \) is a parameter to trade off the two terms in the objective function. It can be shown that \( \text{tr}(K) = 1/(2n) \sum_{i} \sum_{j} ||\Phi_i - \Phi_j||^2 \), i.e., \( \text{tr}(K) \) is proportional to the variance of data in the feature space. We promote low dimensional embedding by adjusting \( \eta \).

To embed the data from a small number of DTW distances, we use OSE. We randomly select \( m (m < n) \) samples from \( n \) time series data in the DB, and apply NPE to 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† and ISOLET††), we compare our method with multidimensional scaling (MDS) [19]. We use the Nyström method [20] as an out-of-sample extension for MDS.

We adjust \( \eta \) in Eq. (2) so as to embed the data in a low dimensional space. Figure 1 shows the eigenvalue distribution for ASL when \( \eta \) 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. 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.

Figure 2 shows the RP curves for the ASL and ISOLET data. In a similarity search, it is important to find almost all of the nearest neighbors. Therefore, we compare the precision results at the point where the recall percentage is very high. Table 3 shows the precision results for ASL and ISOLET at the 90% recall point. The table also shows the number of nearest neighbors to search for in the embedded space to obtain 90% of the 10 nearest neighbors. For both ASL and ISOLET, the NPE+OSE results are better than those for MDS+Nyström. We attribute this to the fact that NPE constructs the kernel using only neighborhood distances, and has no negative eigenvalues.

To examine the effect of the neighborhood size, we also

| dataset | \( K^\text{NPE} \) | \( K^\text{OSE} \) |
|---------|----------------|----------------|
| UNIPEN-1 | NPD | CNE | RNE | NPD | CNE | RNE |
| UNIPEN-2 | 6.0 | 5.2 | 5.6 | 4.8 | 4.8 | 4.8 |

| dataset | \( k \) | \( k \) | \( k \) | \( k \) | \( k \) | \( k \) | \( k \) | \( k \) | \( k \) |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| #1      | 6     | 8     | 12    | 15    | 20    | 50    | 80    | 150   | 250   |
| #2      | 6.8   | 6.4   | 10.0  | 6.4   | 13.6  | 16.8  | 12.8  | 20.8  |

Table 1: LOO-errors for UNIPEN. The error rates for NPD, CSE, RNE, 1-nn, and k-nn are from [14]. 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 \( K^\text{NPE} \) is 3 for both datasets. The value of \( y \) for \( K^\text{OSE} \) is 1.0 except for Tra in dataset #2, where it is 0.75.

Table 2: LOO-errors for UNIPEN with k-nn neighborhoods (6 \( \leq k \) \( \leq 250 \)). All errors are computed by linear SVM with NPE.
The eigenvalue distribution of the kernel matrix for the ASL sample data. The contribution rate $c$ under the embedding dimension $p$, $c = \sum_{i=1}^{p} \lambda_i / \sum_{j=1}^{m} \text{tr}(K)$ is also shown. As $\eta$ 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. (10) and (7), used an $\epsilon$ neighborhood. The value of $\epsilon$ 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 these as DB and as queries. The 28-dimensional feature vector consists of 14 MFCCs and their first-order time derivatives.

6. Conclusion and Future Work

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 introduced NPE, an SDP formulation, to obtain a kernel matrix that best preserves the local geometry of the time series data, together

Contrary to our expectation, larger neighborhood size generally leads to better results. It seems that low dimensional embedding is difficult with small neighborhood.
Table 3. ASL and ISOLET: the rate of $P$ at the point where $N = 90\%$ and the number of nearest neighbors searched to find 9 out of 10 nearest neighbors.

| embedding methods | dim | ASL | ISOLET |
|-------------------|-----|-----|--------|
|                   |     | $P$ (%) | number of NNs searched | $P$ (%) | number of NNs searched |
| NPE+OSE           | 10  | 13.9 | 65     | 17.3   | 53      |
|                   | 20  | 17.3 | 53     | 28.2   | 32      |
| MDS+Nyström       | 10  | 7.26 | 124    | 9.79   | 92      |
|                   | 20  | 7.20 | 125    | 16.4   | 67      |

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.

with its out-of-sample extension. To validate our approach, we have presented two applications: time series classification and time series embedding for similarity search.

In the present work, the test data is limited, particularly for large margin classification. In future work, we aim to use different kinds of time series data, as well as larger-sized data to show the robustness of our method.

References

[1] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.
[2] C. Corres and V. Vapnik, “Support vector networks,” Mach. Learn., vol.20, pp.273–297, 1995.
[3] B. Schölkopf, A. Smola, and K. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” Neural Comput., vol.10, pp.1299–1319, 1998.
[4] L. Rabiner and B. Juang, Fundamentals of Speech Recognition, Prentice Hall, 1993.
[5] H. Shimodaira, K. Noma, M. Nakai, and S. Sagayama, “Dynamic time-alignment kernel in support vector machine,” in Neural Information Processing Systems 14, pp.921–928, MIT Press, 2002.
[6] C. Bahlmann, B. Haasdonk, and B. Burkhardt, “On-line handwriting recognition with support vector machines-a kernel approach,” Proc. 8th Int. W/S on Frontiers in Handwriting Recognition, pp.49–54, 2002.
[7] L. Vandenberghe and S. Boyd, “Semidefinite programming,” SIAM Rev., vol.38, no.1, pp.49–95, 1996.
[8] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan, “Learning the kernel matrix with semidefinite programming,” JMLR, vol.5, pp.27–72, 2004.
[9] K.Q. Weinberger, F. Sha, and L.K. Saul, “Learning a kernel matrix for nonlinear dimensionality reduction,” Proc. 21st Int. Conf. on Machine Learning (ICML-04), pp.839–846, 2004.
[10] F. Lu, S. Keles, S. Wright, and G. Wahba, “Framework for kernel regularization with application to protein clustering,” PNAS, vol.102 no.35, pp.12332–12337, 2005.
[11] A. Hayashi, Y. Mizuhara, and N. Suematsu, “Embedding time series data for classification,” Proc. Int. Conf. Machine Learning and Data Mining, MLDM2005, pp.356–365, 2005.
[12] A. Hayashi, K. Nisizaki, and N. Suematsu, “Fast similarity search of time series data using the nystrom method,” ICDM 2005 Workshop on Temporal Data Mining, pp.157–164, 2005.
[13] J.B. Kruskall and M. Liberman, “The symmetric time warping algorithm: From continuous to discrete,” in Time Warps, String Edits and Macromolecules, Addison-Wesley, 1983.
[14] B. Haasdonk and C. Bahlmann, “Learning with distance substitution kernels,” Proc. 26th DAGM Symp. (DAGM04), pp.220–227, 2004.
[15] K. Toh, R. Tütüncü, and M. Todd, “Solving semidefinite-quadratic-linear programming using sdpt3,” Mathematical Programming, vol.95, pp.180–217, 2003.
[16] J. Friedman, J. Bentley, and R. Finkel, “An algorithm for finding the best matches in logarithmic expected time,” ACM Trans. Math. Softw., vol.3, no.3, pp.209–226, 1977.
[17] W. Kadous, “Australian sign language data in the uci kdd archive,” 1995. http://www.cse.unsw.edu.au/~waleed/ml/data/
[18] R. Cole, Y. Muthusamy, and M. Fanti, “The ISOLET spoken letter database,” Technical Report CS/E 90-004, 1990.
[19] T. Cox and M. Cox, Multidimensional Scaling, Chapman and Hall, 2001.
[20] Y. Bengio, P. Vincent, and J. Paiement, “Learning eigenfunctions links spectral embedding and kernel pca,” Neural Comput., vol.16, no.10, pp.2197–2219, 2004.
Appendix

A.1 SQLP

To apply SDPT-3 [15], a conic programming solver, we introduce Semidefinite-Quadratic-Linear Programming (SQLP) as follows.

SQLP

\[
\begin{align*}
\min & \quad \sum_{j=1}^{n} e_{i_j}^T x_{i_j}^T + \sum_{i=1}^{n} e_{i}^T x_{i}^T + \langle e^n, x^n \rangle \\
\text{s.t.} & \quad \sum_{j=1}^{n} a_{i, j}^T x_{i_j}^T + \sum_{i=1}^{n} a_{i, m}^T x_{i}^T + \langle a_m, x_m \rangle = b_m (1 \leq m < M), \\
& \quad x_j^T K_j^m, x_j^T K_q^m, x_j^T K_q^m, x_j^T K_q^m, x_j^T K_q^m \in \mathbb{R}^{n_j},
\end{align*}
\]

Here, \(e_i^T, x_i^T\) are symmetric matrices of dimension \(s_j\) and \(K_q^m\) is the cone of positive semidefinite symmetric matrices of the same dimension. Similarly, \(e_i^T, x_i^T\) are vectors in \(\mathbb{R}^{n_i}\) and \(K_q^m\) is the quadratic or second-order cone defined by \(K_q^m \triangleq \{ x = [x_0; \bar{x}] \in \mathbb{R}^{n} | x_0 \geq \sqrt{\bar{x}} \}\). Finally, \(e_i^T, x_i^T\) are vectors of dimension \(n_i\), \(K_q^m\) is the nonnegative orthant \(\mathbb{R}^{n_i}_{\geq 0}\), and \(e_i^T, x_i^T\) are vectors of dimension \(n_j\). \(a_{i, j}^T \in S^{n_j}\) is a constraint matrix associated with the \(j\)th semidefinite block variable \(x_j^T\). The vector \(a_{i, m}^T\) is a \(q_i\) dimensional constraint vector corresponding to the \(i\)th quadratic block variable \(x_i^T\), and \(a_{i, m}^T\) are \(n_i\) and \(n_j\) dimensional constraint vectors corresponding to the linear block variable \(x_i^T\) and the unrestricted block variable \(x_j^T\).

A.2 NPE Formulation to a SQLP

NPE formulation is as follows.

\[
\begin{align*}
\min & \quad \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} \sum_{i \in \mathbb{N}} w_{ij} d^2(X_i, X_j) - \langle B_{ij}, K \rangle \\
\text{s.t.} & \quad \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} K(i, j) = 0,
\end{align*}
\]

Let \(N\) be \(\{(i, j) | 1 \leq i \leq n, X_i \sim X_j\}\). To convert \(A.1\) to a SQLP form, we introduce non-negative variables \(u_{ij}^+ \geq 0, u_{ij}^- \geq 0 (i, j) \in N\) and \(u_{ij}^\top u_{ij}^- \geq 0 (i, j) \in N\) to take care of the absolute term in the objective function \(A.1\), as follows.

\[
\begin{align*}
\min & \quad \sum_{k \in \mathbb{N}} \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} w_{ij}^+ (u_{ij}^+ + u_{ij}^-) \\
\text{s.t.} & \quad u_{ij}^\top u_{ij}^\top = d^2(X_i, X_j) - \langle B_{ij}, K \rangle, (i, j) \in N \\
& \quad \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} K(i, j) = 0,
\end{align*}
\]

We stack both \(\{u_{ij}^+ | (i, j) \in N\}\) and \(\{u_{ij}^- | (i, j) \in N\}\) into a vector \(u \in \mathbb{K}^{2|N|}_{\geq 0}\). We also stack \(\{w_{ij} | (i, j) \in N\}\) into a corresponding vector \(w \in \mathbb{R}^{2|N|}\) such that \(\langle w, u \rangle = \sum_{(i, j) \in N} w_{ij} (u_{ij}^+ + u_{ij}^-)\) holds. Then we obtain a SQLP form.

\[
\begin{align*}
\min & \quad \sum_{k \in \mathbb{N}} \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} w_{ij} d^2(X_i, X_j) - \langle B_{ij}, K \rangle \\
\text{s.t.} & \quad \langle B_{ij}, K \rangle + \langle e_{ij}, u \rangle = d^2(X_i, X_j), (i, j) \in N \\
& \quad \langle \text{ones}, K \rangle = 0,
\end{align*}
\]

where \(e_{ij}\) denotes a vector in \(\mathbb{R}^{2|N|}\) such that \(\langle e_{ij}, u \rangle = u_{ij}^+ - u_{ij}^-\) holds, and \(\text{ones}\) is a matrix whose elements are all 1

A.3 OSE Formulation to a SQLP

OSE formulation is as follows.

\[
\begin{align*}
\min & \quad \sum_{i, j \in \mathbb{N}} w_{ij} d^2(X_i, X_j) + \langle B_{ij}, K \rangle \\
\text{s.t.} & \quad K_{n+1} = \begin{bmatrix} K_n & b \\ b^T & c \end{bmatrix} \\
& \quad b = -\langle B_{i, n+1}, K_{n+1} \rangle.
\end{align*}
\]

Note that the following Lemma holds.

\textbf{Lemma 1:}

\[
\begin{align*}
K_n \geq 0, \begin{bmatrix} K_n & b \\ b^T & c \end{bmatrix} \geq 0 \\
\iff K_n \geq 0, b = K_n b, c - b^T K_n^2 b \geq 0
\end{align*}
\]

Proof (sketch): (⇒) Since \(f(x, y) = \langle x, K_n x \rangle + 2 \langle x, b \rangle y + c y^2 \geq 0\) for \(\forall x, \forall y\), we substitute \(x = b - K_n b, y = -1/c|x|^2\) and obtain \(x = 0\). To obtain the last inequality, substitute \(x = K_n b, y = -1\). (⇐) Since \(K_n \geq 0, \langle x_i, K_n x_j \rangle \leq \langle x_i, x_j \rangle \langle x_i, x_j \rangle\) holds for \(\forall x, \forall y\). Then, we can show \(\langle x, K_n b \rangle^2 \geq \langle x, K_n b \rangle\langle K_n b \rangle\) for \(\forall x\). Using this and \(c - b^T K_n^2 b \geq 0\), we can show \(f(x, y) \geq 0\) for \(\forall x, \forall y\). □

We denote the eigen-decomposition of the kernel matrix \(K_n\) as follows.

\[
K_n = U \Lambda U^T,
\]

where \(\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_p), \lambda_1 \geq \ldots \lambda_p > 0\) is a diagonal matrix of the positive eigenvalues, and \(U = [e_1, \ldots, e_p]\) is a matrix of the eigenvectors. We then show

\[
\begin{align*}
K_n & \geq 0, b = K_n b, c - b^T K_n^2 b \geq 0 \\
\iff K_n & \geq 0, \exists \bar{b} \in \mathbb{R}^p, \exists \bar{c} \in \mathbb{R} \ni \bar{b} = U \Lambda^{1/2} \bar{b}, c \geq \bar{c}^2, \bar{c}^2 \geq b^T \bar{b} \\
\iff K_n & \geq 0, \exists \bar{c}, \bar{b}^T \bar{b} \in K_q^m + 1, \\
& \quad \bar{b} = U \Lambda^{1/2} \bar{b}, \begin{bmatrix} 1 & \bar{c} \\ \bar{c} & c \end{bmatrix} \geq 0
\end{align*}
\]

Hence, \(A.2\) can be converted to

\[
\begin{align*}
\min & \quad \sum_{i, j \in \mathbb{N}} w_{ij} d^2(X_i, X_j) + \langle B_{ij}, K \rangle \\
\text{s.t.} & \quad \langle B_{i, n+1}, K_{n+1} \rangle
\end{align*}
\]
\[ s.t. \quad Z = \begin{bmatrix} 1 & \tilde{c} \\ \tilde{c} & c \end{bmatrix}, \quad b = U^{1/2} \tilde{b}, \quad \tilde{K}_{n+1} = \begin{bmatrix} K_n & b \\ b^T & c \end{bmatrix} \]

This can be easily converted to a SQLP by taking care of the absolute values in the objective function and expressing the constraints as linear ones.

---

**Hiroyuki Narita** received a BE degree in Computer Engineering from Hiroshima City University, Hiroshima, Japan, in 2006. Currently, he is a master-course student in the Graduate School of Information Sciences, Hiroshima City University, Hiroshima, Japan. His research interests are in the area of machine learning and pattern recognition.

---

**Yasumasa Sawamura** received the BE and MS degrees in Computer Engineering from Hiroshima City University, Hiroshima, Japan, in 2006. Currently, he is with NEC Aerospace Systems, Ltd.

---

**Akira Hayashi** received a BS degree in Mathematics from Kyoto University, Kyoto, Japan, in 1974. He joined IBM Japan, Ltd., in April, 1974. He received a MS degree in Computer Science from Brown University, Providence, RI, in 1988, and a PhD degree in Computer Science from the University of Texas at Austin, Austin, TX, in 1991. He was a visiting associate professor at the Kyushu Institute of Technology, Fukuoka, Japan, until March 1994. Currently, he is a professor in the Faculty of Information Sciences, Hiroshima City University, Hiroshima, Japan. His research interests are in the area of machine learning and pattern recognition.