Twin Kernel Embedding with Back Constraints

Yi Guo, Paul W. Kwan
School of Science and Technology
University of New England
Armidale, NSW 2351, Australia
{yguo4,kwan}@turing.une.edu.au

Junbin Gao*
School of Acc & Computer Science
Charles Sturt University
Bathurst, NSW 2795, Australia
jbgao@csu.edu.au

Abstract

Twin Kernel Embedding (TKE) is a novel approach for visualization of non-vectorial objects. It preserves the similarity structure in high-dimensional or structured input data and reproduces it in a low dimensional latent space by matching the similarity relations represented by two kernel gram matrices, one kernel for the input data and the other for embedded data. However, there is no explicit mapping from the input data to their corresponding low dimensional embeddings. We obtain this mapping by including the back constraints on the data in TKE in this paper. This procedure still emphasizes the locality preserving. Further, the smooth mapping also solves the problem of so-called out-of-sample problem which is absent in the original TKE. Experimental evaluation on different real world data sets verifies the usefulness of this method.

1 Introduction

Recently, there has been a great deal of interests in dimensionality reduction and manifold learning motivated by the belief that high dimensional data often lie on or near a low dimensional and curved manifold. The benefits of dimensionality reduction include discovering meaningful underlying structures, reducing computational complexity, data visualization and representation. A number of linear and nonlinear methods have proliferated rapidly in recent years. Among them, the methods that favor local information reveal the intrinsic structure of the data and therefore have drawn much attention from researchers in the past few years.

There are several techniques for implementing the local information preserving idea in dimensionality reduction. One common method is to construct a neighborhood graph capturing either the distance/dissimilarity or proximity/similarity information such as Laplacian Eigenmaps (LE) [1]. Another approach is to introduce the so-called back constraints [9] on the input data and their corresponding lower dimensional representatives (embeddings) such as in Locality Preserving Projections (LPP) [8], Neighborhood Preserving Embedding (NPE) [7] and GPLVM with back constraints (BCGP) [9]. The advantages of the back constraints are twofold: firstly, they define a smooth mapping from data to embeddings that preserves the locality; secondly and most attractively, it enables the algorithm to handle new input data and thus solve the so-called out-of-sample problem [2] “free of charge” because the mapping itself defines the corresponding relationship between input data and embeddings.

The techniques mentioned above can be applied to existing algorithms to endow them the ability to handle novel data. Twin Kernel Embedding (TKE) [6] is a newly developed algorithm for non-vectorial data visualization and manifold learning. However, it can not deal with the new data. Followed from the earlier discussion, the out-of-sample problem of TKE can be solved by introducing back constraints into the original TKE algorithm. To facilitate the explanation, we first review those methods with back constraints and then derive the objective function taking constraints into account upon analyzing the original TKE algorithm. The experimental results on real world data sets will be used to support the usefulness of this method and finally we draw our conclusions.

2 Brief Review of Back Constraints in Dimensionality Reduction Algorithms

The following notations will be adopted throughout this paper. The data in the input space are denoted by \{y_i\}_{i=1}^N while vectors \{x_i\}_{i=1}^N their embeddings in a low-dimensional space or the so-called latent space for \(x_i \in \mathbb{R}^d\). Note that in TKE, it is not necessary for \(y_i\)’s to have vectorial form. \(Y\) and \(X\) will be used to denote respectively the set of input objects and the set of corresponding embeddings. If the objects were vectorial, both \(Y\) and \(X\) would

*The author to whom all the correspondences should be addressed.
denote matrices consisting of rows of vectors.

Locality Preserving Projections (LPP) is based on Laplacian Eigenmaps (LE). Back constraints were imposed on the data and their embeddings as

$$x_i = A^\top y_i, \quad \text{(1)}$$

where $A$ is the so-called transformation matrix. This relation is integrated into the objective function of LE by replacing every $x_i$ by (1) and the optimization problem turns to finding an optimal transformation matrix $A$ instead of $X$. Neighborhood Preserving Embedding (NPE) incorporated the same form of back constraints into LLE by using the same substitution technique. Both NPE and LPP exhibit locality preserving ability because the constraints in form of equation (1) implies that the (dis)similarity between input data will be equally reflected in their images in the latent space by the transformation matrix which is actually a linear mapping, or in other words, if $y_i$ and $y_j$ are close, $x_i$ and $x_j$ will stay close as well.

In GPLVM with back constraints (BCGP), more complex back constraints,

$$x_{ij} = \sum_{m=1}^{N} \alpha_{mj} \exp \left( -\frac{\gamma}{2} \| y_i - y_m \|^2 \right), \quad \text{(2)}$$

were used where $\gamma$ is the length scale of the RBF function, $x_{ij}$ is the $j$-th dimension of $x_i$, and $\alpha_{mj}$'s are the parameters to be determined. Again the back constraints are substituted back to the objective function of GPLVM to optimize the $\alpha_{mj}$ instead of $X$.

From the discussion above, we can define the back constraints in general as

$$x_i = f(y_i) + \epsilon_i, \quad \text{(3)}$$

where $f$ is any kind of smooth functions which we call backward mapping function and $\epsilon_i$ an error which can be zero. It can be incorporated into the objective function of host dimensionality reduction algorithms given that the optimization problem is still solvable with respect to the parameters of the constraints after the substitution.

## 3 Twin Kernel Embedding

Twin Kernel Embedding (TKE) preserves the similarity structure of input data in the latent space by matching the similarity relations represented by two kernel Gram matrices, i.e. one for input data and the other for embedded data. It simply minimizes the following objective function with respect to $x_i$'s

$$L = -\sum_{ij} k_x(x_i, x_j)k_y(y_i, y_j)$$

$$+ \lambda_k \sum_{ij} k_x^2(x_i, x_j) + \lambda_x \sum_i \| x_i \|^2, \quad \text{(4)}$$

where $k_x(\cdot, \cdot)$ is the kernel function on embedded data and $k_y(\cdot, \cdot)$ the kernel function on input data. The first term performs the similarity matching mainly. The second and third terms are regularization to control the norms of the kernel and the embeddings. $\lambda_k$ and $\lambda_x$ are tunable positive parameters to control the strength of the regularization.

The conjugate gradient (CG) algorithm can be adopted to get the optimal $X$ for lacking of closed form solution. The hyper-parameters of the kernel function $k_x(\cdot, \cdot)$ (it is RBF kernel normally) can be optimized as well in the minimization procedure. The initial state is given by KPCA or KLE [5].

It is worth explaining the method of locality preserving in TKE. This is done by the $k$ nearest neighboring. Given an object $y_i$, for any other input $y_j$, $k_y(y_i, y_j)$ will be artificially set to 0 if it is not the $k$ largest ones. The parameter $k (> 1)$.

An elegant feature of TKE is that it can handle non-vectorial data since in its objective function, it involves only the kernels that can accept non-vectorial inputs. Through TKE, any kind of data can be visualized in lower dimensional space as long as an appropriate kernel is defined on them.

## 4 Twin Kernel Embedding with Back Constraints

As we can see from the objective function of TKE and its optimization procedure, the out-of-sample problem poses a great challenge to this algorithm in that there is no relation between input data and embedded data at all. From the discussion above, we believe it is possible to incorporate back constraints in TKE to address this problem without sacrificing its original effectiveness in two respects: locality preserving and non-vectorial objects applicability.

We introduce the back constraints using kernel mapping as follows

$$x_{ij} = \sum_{m=1}^{N} \alpha_{mj} k_y(y_i, y_m) \quad \text{(5)}$$

which defines a family of smooth functions as clarified in [11]. So it will satisfy the locality preserving [9] and non-vectorial data applicability. We substitute equation (5) back in the objective function of TKE (equation (4) in section 3) replacing every $x_i$ and minimize it with respect to the parameters in the constraints, namely the $\alpha_{mj}$'s.

Formally, denote $A$ the matrix of $\alpha_{mj}$'s and $K_x$ the kernel Gram matrix derived from $k_y(\cdot, \cdot)$, equation (5) can be written in matrix form

$$X = K_y A \quad \text{(6)}$$

Given new data $Y_N$, the prediction of corresponding new
embeddings $X_N$ is computed efficiently by

$$X_N = K_N A$$  \quad (7)$$

where $K_N = k_y(Y_N, Y)$ and $k_y(Y_N, Y)$ is the short notation for $|Y_N| \times |Y|$ kernel Gram matrix evaluated at all pairs of points from set $Y_N$ and $Y$ and $|Y_N|$ is the cardinality of set $Y_N$.

Here we concisely conclude the above idea of incorporating kernel mapping as back constraints in TKE as BC-TKE algorithm as follows.

**BCTKE algorithm**: Given input data $Y$ and the kernel $k_y(\cdot, \cdot)$, this algorithm will produce the optimal $X$ and $A$.

**Input**: $N$ by $N$ Kernel Gram matrix $K_y$.

**Parameters**: $\lambda_1$, $\lambda_2$, $k$ and target dimension $d$.

**Outputs**: $X$, $A$ (or $\alpha_{mj}$’s).

**Procedure**:

- Initialization: use KPCA or KLE if input are non-vectorial to have initial $X$; give start value to hyperparameters of the kernel $k_x(\cdot, \cdot)$.
- Filtering: Use $k$NN to filter $K_y$ (see the description about TKE in last section).
- Optimization: Minimize $L$ (4) by CG iteratively until it reaches certain termination condition.
- Projection: Substitute the optimal $A$ obtained from the last step into (6) to get the final embeddings.
- Prediction: For new inputs, compute $K_N$ with $k_y(\cdot, \cdot)$ and then use (7) for prediction of new embeddings.

Two remarks: First, the termination condition could be preassigned maximum iterations or the threshold for the consecutive update which are normally used in gradient based optimization methods. Second, the dimension of the embeddings can be estimated by other methods such as [3] for vectorial data. But for non-vectorial objects, further investigation is needed which is one of our future work.

### 5 Experimental Results

Experiments were conducted on real world data sets to demonstrate the effectiveness of the BCTKE when compared with the results of TKE and other methods such as those mentioned in introduction. The ability of handling new data in this algorithm is also shown in the experiments. Finally, an interesting example of image manifold learning is presented as well.

#### 5.1 BCTKE on MNIST Handwritten Digits

A subset of handwritten digits images is extracted from the MNIST database (available at http://yann.lecun.com /exdb/mnist/) consisting of 500 images with 50 images per digit. All images are $28 \times 28$ in grayscale. In order to demonstrate the ability to handle non-vectorial data, we use shape context based IGV (SCIGV) kernel [4] for the given images as $k_y(\cdot, \cdot)$ which is also used in [5] and [6]. The hyper-parameters of the SCIGV kernel are set to the optimal values as reported in the original paper.

We visualize the images in the subset in 2 dimensional Euclidean space. The regularization parameters are set to be 0.005 and 0.001 for $\lambda_1$ and $\lambda_2$, respectively and $k = 13$. The initialization of $X$ is done by KLE.

Because of the absence of an accepted universal evaluation standard for different dimensionality reduction methods, we employ the 1 nearest neighbor (1NN) classification errors to quantify the results of the embeddings generated by different approaches. The smaller the number of errors, the better the method. This evaluation method is also applied in [9]. All the results presented here are the best in the sense of the 1NN classification errors by repeating the experiments in the same setting varying only the parameters of the algorithms, for example the $k$ in LPP and NPE. The best 1NN classification errors of different algorithms are collected in Table 1 for comparison. Apparently, BCTKE has the best score from this point of view.

#### 5.2 BCTKE for New Data

We still use the MNIST handwritten digits mentioned in last subsection. Training set $Y_T$ contains 300 images (30 images per digit) and test set $Y_N$ has 200 images (20 images per digit) so that we can use the mapping function learned from the training set to predict the embeddings for the test data.

We use BCTKE with $K_T (K_T = k_y(Y_T, Y_T)$ where $k_y(\cdot, \cdot)$ is still the SCIGV kernel) to learn the optimal kernel mapping parameters $A$ and predict $X_N$ by using equation (7). Here $K_N = k_y(Y_N, Y_T)$ and $X_N = K_N A$. The parameters of BCTKE are set as in last subsection and the initialization is also provided by KLE. We also notice that the BCTKE is not sensitive to the choice of the parameters as long as the minimization can be carried on without early stop. The parameters of the BCTKE remained the same unless mentioned explicitly.

The results are plotted in Figure 1. From the result of BCTKE on test set we can see clearly that the embeddings

<table>
<thead>
<tr>
<th>Alg</th>
<th>TKE</th>
<th>BCTKE</th>
<th>KLE</th>
<th>GPLVM</th>
<th>LPP</th>
<th>NPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Err</td>
<td>113</td>
<td>107</td>
<td>194</td>
<td>138</td>
<td>219</td>
<td>226</td>
</tr>
</tbody>
</table>

Table 1: Comparison of 1-nearest neighbor classification errors of different algorithms. BCTKE outperforms others from this point of view.
Table 2: Comparison of 1NN classification errors of different algorithms in prediction of the embeddings for new data. BCTKE is the best and BCGP has comparable performance.

<table>
<thead>
<tr>
<th>Alg</th>
<th>BCTKE</th>
<th>BCGP</th>
<th>LPP</th>
<th>NPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union</td>
<td>206</td>
<td>210</td>
<td>303</td>
<td>309</td>
</tr>
<tr>
<td>Train</td>
<td>79</td>
<td>86</td>
<td>133</td>
<td>143</td>
</tr>
<tr>
<td>Test</td>
<td>103</td>
<td>110</td>
<td>154</td>
<td>164</td>
</tr>
</tbody>
</table>

of the new data still have relatively distinguishable clusters and the result on union set reveals the smoothness of the mapping function since the new embeddings located around the clusters taken by the existing embeddings. BCGP has a comparable result with BCTKE from both the visual observation and 1NN classification errors (see table 2) because nonlinear mapping functions are adopted and learned in BCTKE and BCGP. While the linear mapping function (transformation matrices) in LPP and NPE fails to capture and highly nonlinear relation between input data and their corresponding embeddings which is demonstrated by the large amount overlapping existing in their results on test set.

5.3 Text Classification Comparison

To further compare the performance of BCTKE with other methods on different data sets, they are applied to the classification on texts where the data set is Reuters-21578 Text Categorization Test Collection (Distribution 1.0 is available from http://www.daviddlewis.com/resources/test collections/reuters21578). Training set is composed of 10 arbitrarily selected topics extracted from TOPICS category in Reuters-21578 database with 30 pieces of news in each topic while test set consist of 20 pieces in each topic.

The kernel used in BCTKE is string subsequence kernel (SSK) [10] which simply treats the text data as strings of characters. In order to make the texts applicable to other methods depending on vectors, we vectorized the texts by vector space model (VSM) [12]. This experiment is to examine the quality of the prediction for new input data in latent space with different dimensionality which is quantified by classification error rate. The prediction of different algorithms on new input data is based on the model learned from training set without any special optimization for the classification. The procedure is: the dimensionality reduction method will first use the labeled training set to learn the mapping function and project them into target latent space with specified dimension to be known patterns, and then predict the embeddings of the test data using the mapping function to be unknown patterns into the latent space where a naive classifier kNN will be carried out to classify them. Since the labels of the test data are also provided, we can obtain the classification error rate.

The target latent space ranges from 1D to 25D and the kNN classifiers with different k (from 1 to 20) are tested for each case whence the mean and the standard deviation of classification error rate can be collected which is used for comparison.

The classification error rates of BCTKE and LPP are plotted in Figure 2. The absence of BCGP is caused by the inhibitive computational cost since the time costing matrix inversion operation is involved in its optimization. NPE fails to work because the vectors provided by VSM are extremely sparse and break down the construction of the weight matrix which is pivot of the algorithm. From Figure 2, we can see that when d is small, BCTKE and LPP act quite similarly. But when d grows larger, remarkable difference can be observed from the fact that the BCTKE has average lower classification error rate and smaller standard deviations for different kNN classifiers which informs that the BCTKE produces better prediction for new input data. Note that this experiment is mainly for illustration. The high classification error rates for both methods could probably be optimized further by incorporating supervised learning.

5.4 BCTKE on Image Manifold Learning

Lastly, we present the result of BCTKE on image manifold learning. The objects are 1965 images (each image is 20 × 28 grayscale) of a single person’s face extracted from a digital movie. Since the images are well aligned, we simply use the linear kernel as $k_y(\cdot, \cdot)$ and the BCTKE is initialized by KPCA.

Two facts can be observed from the result shown in Fig-
Figure 1: BCTKE in prediction and comparison with other methods. The results of different algorithms are organized in rows. From top to bottom, they are BCTKE, BCGP, LPP (with $k = 4$) and NPE (with $k = 5$). The columns are the results of different algorithms on different sets. They are training set, test set and the union of training and test set from left to right. All the figures share the same legend as shown in up left corner figure. The results presented here are also optimized with respect to the 1NN classification errors on the union of training set and test set as discussed in last subsection. The errors are collected in table 2.
can be incorporated into other host dimensionality reduc-
tion or manifold learning methods. The mapping functions
can be seen as an applicable component in the family of
dimensionality reduction methods and the integration will
probably lead to new algorithms.

Acknowledgements

This work is supported by the National Natural Science
Foundation of China (NSFC 60373090 ), the ARC DP De-
velopment Grant from Charles Sturt University and the Un-
iversity Research Grant from the University of New Eng-
land.

References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimen-
sionality reduction and data representation. Neural Compu-
and M. Ouimet. Out-of-sample extensions for lle, isomap,
mds, eigenmaps, and spectral clustering. In Advances in
Neural Information Processing Systems 16.
f ormation Processing Systems 15. MIT Press.
semigroup kernel in image classification. In International
maps for visualization of non-vectorial data. In A. Sattar
and B. Kang, editors, Lecture Notes on Artificial Intelli-
gent, volume 4304, pages 1179–1183, Berlin Heidelberg,
December 2006. Springer-Verlag.
vectorial data using twin kernel embedding. AIDM, 0:11–
17, 2006.
[7] X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood pre-
S. Thrun, L. Saul, and B. Schölkopf, editors, Advances in
Neural Information Processing Systems 16. MIT Press,
[9] N. D. Lawrence and J. Quinonero-Candela. Local distance
preservation in the GP-LVM through back constraints. In
W. Cohen and A. Moore, editors, Proceedings of the Inter-
national Conference in Machine Learning, San Francisco,
ses for Machine Learning. Adaptive computation and machine
for automatic indexing. Information Retrieval and Language