Twin Kernel Embedding

Yi Guo, Student Member, IEEE, Junbin Gao, and Paul W. Kwan, Member, IEEE

Abstract—In most existing dimensionality reduction algorithms, the main objective is to preserve relational structure among objects of the input space in a low-dimensional embedding space. This is achieved by minimizing the inconsistency between two similarity/dissimilarity measures—one for the input data and another for the embedded data—via a separate matching objective function. Based on this idea, a new dimensionality reduction method called Twin Kernel Embedding (TKE) is proposed. TKE addresses the problem of visualizing nonvectorial data that is difficult for conventional methods in practice due to the lack of efficient vectorial representation. TKE solves this problem by minimizing the inconsistency between the similarity measures captured respectively by their kernel Gram matrices in the two spaces. In the implementation, by optimizing a nonlinear objective function using the gradient descent algorithm, a local minimum can be reached. The results obtained include both the optimal similarity-preserving embedding and the appropriate values for the hyperparameters of the kernel. Experimental evaluation on real nonvectorial data sets confirmed the effectiveness of TKE. TKE can be applied to other types of data beyond those mentioned in this paper whenever suitable measures of similarity/dissimilarity can be defined on the input data.

Index Terms—Dimensionality reduction, Twin Kernel Embedding, kernel learning.

1 INTRODUCTION

DIMENSIONALITY reduction (DR) is one of the important prepro-
cessing steps in many advanced applications such as exploratory
data analysis and manifold learning. It has been successfully
applied in many areas, including robotics, bioinformatics, etc.
The goal of DR is mainly to find the corresponding mappings of input
data in a much lower dimensional space without incurring
significant information loss. The low-dimensional representation
can be used in subsequent procedures such as classification and
pattern recognition. In the area of machine learning, many well-
known DR methods [2], [7], [10], [15], [18], [20] have been reported
in the literature that can handle different kinds of data and
produce either linear or nonlinear embedding.

In our analysis of these representative methods, we observed
that they share a similar design pattern. That is, the measure that
describes the relational structure among the data in the input space
is matched as closely as possible to that of the latent (or
embedding) space. Such measures could be the degree of
similarity, degree of dissimilarity, or the commonly used euclidean
distance. For example, in Multidimensional Scaling (MDS) [2], the
inner products in both the input and the latent space are matched.
On the other hand, the cost function being optimized in Laplacian
Eigenmaps (LEs) [1] results in embeddings whose pairwise
euclidean distances are obtained from the proximity information
of their corresponding input data.

This observation motivated us to consider other kinds of
measures for the input space, such as a similarity measure used in
bioinformatics [14] or a similarity evaluated by a kernel function
[17] that can handle the so-called nonvectorial data [4].

The same idea can be applied to the latent space by choosing
other measures instead of the euclidean distance, like a kernel
function or another distance metric that has analytical forms
written in terms of the embeddings so that the optimization
problem is solvable. By choosing a nonlinear kernel as a similarity
measure in the latent space, one can endow the algorithm with the
ability to deal with nonlinearity that exists among embedded data
in the low-dimensional space directly. This is a desirable feature
for applications in which the linearity assumption is not valid.

Furthermore, different objective functions can be chosen to
express the design objectives of different DR methods. For MDS,
the objective function conveys that if two objects are close in the
input space, their embeddings should also be close in the latent
space. For LE, it discourages solutions in which the proximity
of input data is high but their corresponding embeddings are far
apart in the latent space. Of course, other forms of objective
functions can be chosen provided that their optimization with
respect to the embeddings is feasible.

Based on the preceding observation and analysis, we propose a
new DR method, called Twin Kernel Embedding (TKE) [8], that
utilizes kernels as measures for both input data and their
corresponding embeddings. Section 2 will describe the TKE and
explain both its design and the optimization procedure. Experi-
mental evaluation on real nonvectorial data sets is presented in
Section 3 to demonstrate the effectiveness of TKE. Finally, we
conclude by discussing potential applications and future work in
Section 4.

2 TWIN KERNEL EMBEDDING

The following notations are used in this paper. The data in the
input space are denoted by \{(\textbf{y}_i, \textbf{y}_j)\}_{i=1}^{N}, while \{(\textbf{x}_i, \textbf{x}_j)\}_{i=1}^{N}
denotes their embeddings in the latent space of dimension \textit{d}. In addition, \textbf{Y}
and \textbf{X} will be used to denote, respectively, the set of input objects and
the set of embeddings. If the objects were vectorial, \textbf{Y} would
denote a matrix with data in rows.

2.1 General Framework

Most DR algorithms can be seen as being composed of three major
components that can be chosen separately: the measure for input
data \textit{M}_\textit{y}, the measure for embeddings \textit{M}_\textit{x}, and the objective
function \textit{L} that realizes the matching. The measures can be chosen
individually in the sense of similarity or dissimilarity, while \textit{L}
will be formulated to minimize the inconsistency between the two
measures accordingly. The supplement of this paper, which can be
found in the Computer Society Digital Library at http://
doi.ieee.org/10.1109/TPAMI.2008.74, analyzes several well-known DR methods from this point of view.

2.2 Kernels on Data and Similarity Matching

As pointed out in [4], a kernel can be regarded as a similarity
measure for nonvectorial data (text, proteins, gene expressions,
etc., which are not readily vectorized and frequently appear in
machine learning). Furthermore, from images to DNA sequences, a
variety of kernels have been proposed recently which closely
reflect the domain knowledge [3], [19], [22]. We therefore select a
kernel as the measure for input data, which is denoted by \textit{k}_\textit{y}(\cdot, \cdot).

For the embeddings, we choose the \textit{homogeneous} kernel as the
similarity measure. Typically, it is an RBF kernel
\[
\textit{k}_\textit{x}(\textbf{x}_i, \textbf{x}_j) = \gamma \exp \left(-\sigma ||\textbf{x}_i - \textbf{x}_j||^2 \right),
\]
Because kernels are used in both the input and the latent spaces, we simply need to make them as consistent as possible. Hence, we maximize the following objective function written as the inner product of two vectors expressing the similarities:

\[ C_o = \frac{\text{Vec} K_x \text{Vec} K_y}{||\text{Vec} K_x|| ||\text{Vec} K_y||} = \frac{\text{tr} K_x K_y}{\text{tr}(K_x^2 \text{tr}(K_y^2)),} \tag{2} \]

where \( K_x \) denotes the Gram matrix induced by \( k_x(\cdot, \cdot) \) and \( K_y \) denotes the Gram matrix induced by \( k_y(\cdot, \cdot) \). This is equivalent to minimizing the angle between two vectors \( \text{Vec} K_x \) and \( \text{Vec} K_y \). If they perfectly match, the cosine angle should be 0, indicating that the two similarity vectors are parallel and, thus, \( K_x \) is a constant multiple of \( K_y \). In other words, the similarity relations among pairs of objects (in the kernel sense) will be mapped to their corresponding embeddings in the latent space. As a result, the similarity structure would be well preserved in the latent space.

One should notice that the matching of similarity measures in (2) is based on the inner product, which can also be interpreted as a linear kernel. In other words, it is actually evaluating the similarity of two normalized vectors. Thus, the greater the value of \( C_o \), the more similar these two vectors become. Likewise, it is possible to minimize the difference between these two similarity measures like that of the stress function used in MDS:

\[ C_o = \sum_{ij} (k_x(y_i, y_j) - k_x(x_i, x_j))^2. \tag{3} \]

Other objective functions can also be used. Because a pair of kernels are involved in the formulation of this new algorithm, we call it the Twin Kernel Embedding (TKE).

### 2.3 Optimization

In this section, we shift our focus to the optimization problem defined by (2) with respect to the embeddings and the hyperparameters of the kernel \( k_x \). For computational convenience, we rewrite (2) in a form that incorporates a regularization term:

\[ C'_o = -\sum_{ij} k_x(x_i, x_j) k_y(y_i, y_j) + \lambda_k \sum_{ij} k_x(x_i, x_j)^2. \tag{4} \]

At the same time, we changed the optimization problem from a maximization to a minimization. Also, we need to put extra constraints on the norm of the embeddings in order to limit the solution space. This is necessary for ruling out the multiple solutions caused by the different scales of embeddings. The new objective function becomes

\[ L = -\sum_{ij} k_x(x_i, x_j) k_y(y_i, y_j) + \lambda_k \sum_{ij} k_x(x_i, x_j)^2 + \lambda_n \sum_i x_i^2, \tag{5} \]

where \( \lambda_k > 0 \) and \( \lambda_n > 0 \) are tunable parameters for controlling the strength of the regularizations while all of the constants have been discarded. The objective function \( L \) can be rewritten in matrix form as

\[ L = -\text{tr}(K_x K_y) + \lambda_k \text{tr}(K_x K_x) + \lambda_n \text{tr}(XX^T). \tag{6} \]

Because our targets, i.e., \( x_i \)'s, are embedded in a nonlinear kernel function, the optimization is generally not convex. As a result, (6) has no closed-form solution. One has to employ a gradient descent-based optimization procedure on the objective function \( L \). The partial derivative of \( L \) with respect to \( K_x \) is

\[ \frac{\partial L}{\partial K_x} = -K_y + 2\lambda_k K_x. \tag{7} \]

From this, one can obtain the partial derivatives of \( L \) with respect to \( K_y \) and the hyperparameter of \( k_x \) through the chain rule. The scaled conjugate gradient (SCG) algorithm [13] can be used to compute the optimal \( K_x \), \( \gamma \), and \( \sigma \). Similarly, we can complete the optimization of the objective function by using the stress function (3) as

\[ L = \sum_{ij} (k_x(x_i, x_j) - k_y(y_i, y_j))^2 + \lambda_n \sum_i x_i^2 \]

\[ = \text{tr}(K_x^2) - 2\text{tr}(K_x K_y) + \text{tr}(K_y^2) + \lambda_n \text{tr}(XX^T), \tag{8} \]

while noting that it is a special case of (5).

The dimension of the embedded data \( d \) is usually assigned according to the requirements of the application. For example, in visualization, it is normally 2. As a by-product of this optimization process, the optimal \( \gamma \) and \( \sigma \) ensure that the kernel is well adjusted for the task at hand. This motivates us to introduce more complicated kernels for \( X \), such as the mixture of different kinds of kernels that still satisfies the conditions for a valid kernel. The only condition is that the kernel must have an analytic form and be differentiable with respect to its arguments and hyperparameters.

### 2.4 Implementation

Because a gradient descent-based optimization method is applied, an initial value for \( X \) must be assigned. It is also necessary to design a stopping criterion such as the maximum number of iterations or a threshold. The initialization can be obtained by using the result from other DR approaches. If pure nonvectorial data applicability is desirable, KLE [7] and KPCA [18] would be suitable candidates because they depend only on the kernel Gram matrix on the input data. Random initialization is not encouraged due to the nonconvexity of the objective function of TKE.

Some DR methods are known to be locality preserving, such as KLE and LE that pay attention to local information of the input data only. In essence, such a property can easily be incorporated into TKE by performing the \( n \)-nearest neighbor filtering on \( K_x \). For the \( i \)-th row in \( K_x \), only the \( n \) largest entries remain the same, while the others are purposefully set to 0. The variable \( n (>1) \) in \( n \)-nearest neighboring controls the locality that the algorithm will preserve. The reasons for incorporating the filtering procedure are twofold. First, this procedure of choosing the \( n \)-nearest neighbors serves to divide the input data into neighborhoods based on similarity. Therefore, the locality information contained in the input data can be well preserved in the latent space. Second, filtering will relax the constraint of the optimization since those
zeros could mean such pairs are not similar and should be mapped far away without having to provide a rigid condition on how far away they should be. Notice that the initialization methods will still use the original kernel Gram matrix \( K_y \) without filtering since some of them, such as KPCA, require a proper symmetric positive definite kernel Gram matrix.

### 2.5 Some Remarks on TKE

#### Suitable kernels

As measures of similarities, kernels are expected to reflect the intrinsic similarity structure of the data. The more suitable the \( K_y \), the more accurately TKE will reveal the relationships among the input data. However, how to choose an appropriate kernel for \( Y \) is still an open problem. Breakthroughs on this problem by researchers working on kernel theories are anticipated.

**Parameters selection.** Empirical analysis shows that TKE is not sensitive to the selection of the \( \lambda_k \) and \( \lambda_y \), given the SCG searching can be carried on without premature termination. As to the size of the neighborhood for the \( n \)-nearest neighbor filtering, we present experimental results in Section 3.3 to illustrate the effect of this parameter.

**Generic similarity rather than a kernel measure.** One will notice that, in the objective function of TKE (5), no special constraints are imposed on \( K_y \) except for symmetry. In other words, the positive definite property for \( K_y \) is no longer a required condition here. In practice, \( K_y \) is even filtered for the sake of local similarity preserving. It means that generic similarity measures other than kernels for \( Y \) are also applicable in TKE. Similarity measures that reflect domain knowledge such as [14] can be incorporated in TKE directly. Even dissimilarity measures can be included in this algorithm. Some transformations such as exponential function can convert a dissimilarity to similarity. This flexibility extends the range of applicability of TKE.

#### Learning a kernel function

Notice that, on the output side of TKE, an optimal kernel function \( (k_y) \) can be learned from the input data as well as the embeddings \( (X) \). This result reveals the fact that TKE actually offers an approach to learn kernel functions (for example, the RBF kernel) from the similarity matrix. Since this is a valid kernel function, further applications of it in kernel machines are possible. The major difference of TKE to other kernel matrix learning algorithms (kernel alignment score etc.) such as [21] is that TKE will not only obtain the kernel function \( k_y(\cdot, \cdot) \) but also a set of embeddings for the input data. Because TKE defines the type of the kernel function explicitly and learning of its optimal form originates from the similarity matching, it tactically avoids the time-consuming semidefinite programming, which is essential in other kernel learning algorithms to ensure the positive definiteness.

### 2.6 Generalization

A drawback of TKE is that it does not provide the prediction for novel samples. One possible solution is to introduce a smooth mapping function [9] as

\[
x_{ij} = \sum_{m=1}^{N} a_{mj} k_y(y_i, y_m),
\]

which can be substituted into the objective function of TKE (5) by replacing every \( x_i \) and minimizing with respect to the \( a_{mj} \)s accordingly. If we stack \( a_{mj} \)s into a matrix \( A \) as \([A]_{m,j} = a_{mj}\) where \( m = 1 \ldots N, j = 1 \ldots d \), then \( X = K_y A \). After we get \( \frac{\partial F}{\partial A} \) in TKE, it is straightforward to obtain \( \frac{\partial F}{\partial X} \) through the chain rule. Thus, we learn the mapping function instead of the embeddings through which the novel input data can be handled.

3. The initialization of \( A \) can be given by \( K_y^{-1} X \), where \( K_y^{-1} \) is the pseudoinverse of \( K_y \) and \( X \) can be given using other DR methods.

### 3 Experimental Results

Experiments were conducted on real-world data sets to demonstrate the effectiveness of the TKE algorithm when compared with the results of other approaches. For the same data set (MNIST), the result of varying the \( n \) to display the influence of this parameter on TKE is also included in Section 3.3. This result provides a tentative guidance for future applications of TKE. Additional experimental results conducted on other data sets can be found in the supplement, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2008.74.

#### 3.1 Handwritten Digits Visualization

We begin with visualizing handwritten digits. A subset of handwritten digits images is extracted from the MNIST database (available at http://yann.lecun.com/exdb/mnist/). The data set consists of 500 images with 50 images per digit. All images are in gray scale and have a uniform size of \( 28 \times 28 \) pixels. As a result, the input data lie in a space of dimension 784. We use the shape context-based IGV (SCIGV) kernel [6], which is specially designed for shapes in images. The hyperparameters of the SCIGV kernel are set to the optimal values as reported in the original paper. The result of using a simple linear kernel is also presented for comparison.

The latent space is a two-dimensional euclidean space which facilitates human observers interpreting the relational structure of the input data. The RBF kernel (1) was employed as a similarity measure of the latent space in this experiment and the hyperparameters of the RBF kernel will be learned in the optimization process. The regularization parameters are set to \( \lambda_k = 0.005 \) and \( \lambda_y = 0.001 \), respectively, and \( n = 13 \) in \( n \)-nearest neighbor to filter \( K_y \) (we use this set of parameters in subsequent experiments if not mentioned explicitly). We will discuss the selection of \( n \) later. The initial state of \( X \) is given by KLE with the same kernel and \( n \) is set to 3. For hyperparameters of \( k_y \) (\( \gamma \) and \( \sigma \)), we set them to 1 throughout the experiments.

The experimental result is presented in Fig. 1. We also include the results of other methods for comparing the performance on the same data set.1 Visually, the results of TKE are much better than the others. All of the 2D representations by TKE (using SCIGV kernel and linear kernel) reveal clearer clusters of the digits than others. To give a quantitative analysis on the quality of the clusters, we use the “purity” metric [3] as a standard. The purity is defined as

\[
\text{purity} = \frac{1}{N} \sum_{i=1}^{N} \frac{v_i}{n_i}
\]

where \( n_i \) is the number of data points having the same class labels as a given point \( i \) in the \( n \)-nearest neighbors of point \( i \). It uses the fraction of the number of samples from the same class as a given point in a neighborhood with size \( n \) and the purity is the average of the fraction over all points. The higher the purity, the better the clusters. In Fig. 2a, it is shown that the purity of TKE (TKE with linear kernel) is always higher than others, which demonstrates that the clusters produced by TKE are better. To quantify the results further, we use the 1-nearest neighbor (1-NN) classification errors, as described in [11], i.e., the leave-one-out classification errors of 1-NN classifier on the embeddings. The smaller the number of errors, the better the method. The results are collected in Table 1 and the result of each method is the best that it can achieve by choosing the optimal parameters for the method (if any) according to this standard, with the plot shown in Fig. 1 accordingly.

4. Notice that the conventional image vectors are used in other methods as input. See the supplement, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2008.74, for a more comprehensive comparison of TKE with other popular DR methods.
3.2 Visualizing Texts

To further demonstrate its nonvectorial data handling ability, TKE is applied to the Reuters-21578 Text Categorization Test Collection (http://www.daviddlewis.com/resources/testcollections/reuters21578). The test set consists of 10 arbitrarily selected topics and 500 documents (50 documents for each topic) extracted randomly from the TOPICS category in Reuters-21578. The string subsequence kernel (SSK) [12] is employed naturally to evaluate the similarity between texts. We also use the VSM [16] to vectorize these texts for other DR algorithms for comparison. Here, the initial state of TKE is provided by KPCA. The results are presented in Fig. 3, with the topics displayed as the legend in Fig. 3b.

The result of TKE reveals clear cluster structure according to our expectation: The texts from the same topic are expected to be close, while those of others are indistinguishable. An interesting observation is that the results of the texts from the topics cocoa (“o”) and corn (“+”) are almost the same because the corresponding kernel values are very close or equal to 1. In other words, the underlying SSK does not significantly distinguish the topics cocoa and corn. This also indicates that TKE correctly reflects its philosophy, i.e., if two objects are similar in the sense of a given kernel, they will stay close together in the latent space. The comparison on the purity of the clusters produced by these methods is given in Fig. 2b. It objectively shows that the result of TKE with the linear kernel outperforms the others.

Table 1

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Leave-one-out 1-NN Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>TKE (SCIGV)</td>
<td>113</td>
</tr>
<tr>
<td>TKE (Linear)</td>
<td><strong>92</strong></td>
</tr>
<tr>
<td>KLE</td>
<td>127</td>
</tr>
<tr>
<td>LE</td>
<td>155</td>
</tr>
<tr>
<td>LLE</td>
<td>170</td>
</tr>
<tr>
<td>IsoMap</td>
<td>222</td>
</tr>
<tr>
<td>KPCA</td>
<td>292</td>
</tr>
</tbody>
</table>

TKE with the linear kernel outperforms the others.
TKE is the best. Because the vectors generated by VSM have very high dimensionality (6,659) and the input space is extremely sparse, some DR algorithms failed to work in this setting, such as MDS and LLE. This is the reason for their absence in this experiment.

### 3.3 The Influence of $n$ on TKE

Finally, TKE is designed to reflect the local structural information as KLE, LE, etc. The $n$-nearest neighbor is applied to filter the kernel Gram matrix on the input data. In order to examine the influence of $n$, a series of experiments is conducted on the same data set as the one used in Section 3.1 by varying only $n$ while keeping other conditions fixed. KLE is also performed under the same setting. In the set of TKE experiments, KLE with $n = 6$ was used as the initialization.

In each experiment, we recorded the 1-NN classification errors and plot the results in Fig. 4. The blue and red pluses denote the results for TKE with linear kernel and SCIGV kernel, respectively, while the black and pink crosses denote those for KLE with the linear kernel and the SCIGV kernel, respectively. The $n$ ranges from 1 to 60. The minimum error for TKE appears at $n = 11 \sim 13$ and, as $n$ gets larger, the result deteriorates slowly. Meanwhile, the optimum of KLE emerges when $n = 3 \sim 5$ and its error rate increases rapidly as $n$ becomes larger. It turns out that TKE exhibits more tolerance to $n$ in the $n$-nearest neighbor filtering. But, KLE is sensitive to the selection of this parameter and its result deteriorates faster as $n$ increases. Fig. 4 also clearly reveals that TKE is much better than KLE in the sense of the 1-NN classification error.

It is noteworthy that, when $n$ is small, TKE seems to perform contrary to expectation. One possible explanation could be that an $n$ that is too small would cause information loss to the extent that the kernel in the latent space might not capture the similarity structure properly, thereby increasing the degree of uncertainty, and the performance becomes worse than KLE. The same reasoning could be plausible when $n$ is too large since too much information may adversely affect the local information preservation ability of the kernel in the latent space to reproduce the similarity structure. Empirically, we set $n$ to be $10 \sim 15$ or around the number of classes (if known). A priori knowledge on the input data will undoubtedly be beneficial to the selection of this parameter.

### 4 Conclusions

In this paper, a new DR algorithm called TKE has been proposed, in which both measures, i.e., input and latent spaces, are similarities represented by kernel functions. Because of the properties of kernels, TKE is endowed with the nonvectorial data handling ability. As a result, both a valid kernel and the embeddings in the latent space can be learned simultaneously by the TKE process. This characteristic of TKE integrates kernel learning into DR. Not only can metric transformation be realized, but the embeddings of the input objects in the low-dimensional space can also be learned with well-preserved similarity structures.
TKE favors local, rather than global, similarity preservation by filtering the kernel Gram matrix defined over the input data. Due to the choice of the optimization method, the kernels on the embedded data in the lower dimensional space must have analytic form in order to ensure that the derivatives with respect to $X$ is solvable and, hence, the gradients are computable. Since the solution is found by a gradient descent-based optimization process, the initial state must be assigned. KPCA, KLE, or other applicable nonvectorial data methods can be suitable candidates in coming up with such an initial state if input data are highly structured.

Promising applications of TKE include DR, exploratory data analysis, clustering, etc., for both vectorial and nonvectorial objects.

**ACKNOWLEDGMENTS**

The authors would like to thank Dr. Neil D. Lawrence for sharing his GPLVM code, which was partly used in their algorithms, and the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China under Grant NSFC 60373090, by the ARC DP Development Grant from Charles Sturt University, and by a University Research Grant from the University of New England.

**REFERENCES**


