Manifold-preserving graph reduction for sparse semi-supervised learning

Shiliang Sun [a,*] , Zakria Hussain [b] , John Shawe-Taylor [b]

[a] *Department of Computer Science and Technology, East China Normal University, 500 Dongchuan Road, Shanghai 200241, China*

[b] *Department of Computer Science, University College London, United Kingdom*

Abstract

Representing manifolds using fewer examples has the advantages of eliminating the influence of outliers and noisy points and simultaneously accelerating the evaluation of predictors learned from the manifolds. In this paper, we give the definition of manifold-preserving sparse graphs as a representation of sparsified manifolds and present a simple and efficient manifold-preserving graph reduction algorithm. To characterize the manifold-preserving properties, we derive a bound on the expected connectivity between a randomly picked point outside of a sparse graph and its closest vertex in the sparse graph. We also bound the approximation ratio of the proposed graph reduction algorithm. Moreover, we apply manifold-preserving sparse graphs to semi-supervised learning and propose sparse Laplacian support vector machines (SVMs). After characterizing the empirical Rademacher complexity of the function class induced by the sparse Laplacian SVMs, which is closely related to their generalization errors, we further report experimental results on multiple data sets which indicate their feasibility for classification.

Key words: Sparsity; Graph reduction; Support vector machine; Statistical learning theory; Semi-supervised learning

1 Introduction

Learning with the manifold assumption has been an active research topic during the past decade, with a variety of successful applications such as nonlinear dimensionality reduction, data representation, and semi-supervised learn-

* Corresponding author. Tel.: +86-21-54345186; fax: +86-21-54345119.
E-mail address: shiliangsun@gmail.com, slsun@cs.ecnu.edu.cn (S. Sun).

ing [4,19,23,26]. A usual procedure includes constructing a weighted graph using all the training examples and then performing learning based on the graph. However, it has two shortcomings: (1) Possible outliers and noisy points, likely to damage the manifold structure, are retained; (2) The evaluation of predictors learned from the graph for new examples can be time-consuming if the predictors involve computations on all the examples in the original graph.

Although some methods such as random sampling or k-means clustering can be used to reduce the size of the graph, they have no guarantees of preserving the manifold structure or effectively removing outliers and noisy examples. In particular, the k-means method is sensitive to outliers, and time-consuming when the number of clusters is large. To overcome the aforementioned two shortcomings, in this paper we propose the idea of manifold-preserving sparse graphs and the corresponding manifold-preserving graph reduction algorithm, detailed in Section 2 and Section 3, respectively. After providing two case studies to illustrate the performance of the algorithm, we also give some related theoretical outcomes.

As an important application of manifold-preserving sparse graphs, we consider the problem of semi-supervised learning [7,29]. Indeed, a family of recent semi-supervised classification methods build themselves on the exploitation of manifolds, such as Laplacian SVMs [6], co-Laplacian SVMs [22] and manifold co-regularization [23]. They learn a classifier in a reproducing kernel Hilbert space (RKHS) making use of the representer theorem [15]. As a result, the classifier is a function of all labeled and unlabeled examples involved in the training set. Considering the large number of unlabeled examples in semi-supervised learning, kernel function evaluations would be very time-consuming when inferring the label of a new example. Consequently, it is necessary to devise semi-supervised learning methods with a less demanding computational requirement.

In Section 4 we present sparse Laplacian SVMs where manifold-preserving sparse graphs play a central role. By the use of sparse graphs, only a portion of unlabeled examples are needed to evaluate kernel functions. Moreover, outliers and noisy examples are expected to be eliminated by the manifold-preserving graph reduction procedure. This would enhance the robustness of the corresponding algorithm. In Section 5 we derive the empirical Rademacher complexity of the function class induced by sparse Laplacian SVMs, which is an important term in the margin bound of their generalization performance. Experimental results on multiple synthetic and real-world data sets are reported in Section 6 to evaluate the sparse Laplacian SVMs, followed by a conclusion section at the end of this paper.

2 Manifold-preserving sparse graphs

Since graphs are deemed as a discrete representation of manifolds, our definition of manifold-preserving sparse graphs naturally corresponds to sparsified manifolds. To begin with, we present the definition of sparse graph candidates.

Definition 1 (Sparse graph candidates) *Given a graph $G(V, E, W)$ corresponding to a manifold with vertex set $V = \{x_1, \ldots, x_m\}$, edge set E, and symmetric weight matrix W, the graph $G_c(V_c, E_c, W_c)$ with V_c, E_c and W_c being respectively subsets of V, E, and W is called a sparse graph candidate of the original graph G.*

For a graph, the weight on each edge characterizes the similarity or closeness of the linked pair of vertices where a large value corresponds to a high similarity. In this paper, we do not investigate the distinctions of properties of graphs constructed by different methods, but assume that a reasonable graph can be constructed. For the sparse graph candidate G_c, the associated weight matrix W_c is the subset of W defined on selected vertices. This inheritance of weights can ensure the preservation of manifold structures, though to a very limited extent, which is an important concern for manifold sparsification.

We give the following definition of graph distance as a characterization of the loss between two graphs G and G_c. This concept would be beneficial in gauging the degree of sparsification.

Definition 2 (Graph distance) *Given a graph $G(V, E, W)$ and its sparse graph candidate $G_c(V_c, E_c, W_c)$, the graph distance between G and G_c is the loss of weights from W to W_c, that is $\sum_{i \in V \backslash V_c, j \in V_c} W_{ij}$, where $V \backslash V_c$ denotes the set of vertices not included in V_c.*

Suppose we call the percentage loss of weights the *sparsity level*, that is, the above graph distance divided by the sum of weights in graph G. For real applications, it is both common to find sparse graphs with a specific sparsity level or a fixed number of retained vertices (usually these two approaches can be interchangeably adopted).

Consequently, the problem of seeking manifold-preserving sparse graphs would be to find sparse graph candidates with manifold-preserving properties. By manifold-preserving properties, we mean that a point outside of the sparse graphs should have a high connectivity with a point retained in the sparse graphs. Thus, we reach the following definition of manifold-preserving sparse graphs.

Definition 3 (Manifold-preserving sparse graphs) *Given a graph G with m vertices and the sparsity level or the number of vertices in the desired sparse* *graphs, the manifold-preserving sparse graphs G_s are those candidates G_c having a high space connectivity with G. By a high space connectivity, we mean that for a candidate with t vertices the quantity*

$$\frac{1}{m-t}\sum_{i=t+1}^{m}\left(\max_{j=1,\ldots,t} W_{ij}\right) \tag{1}$$

is maximized, where W is the weight matrix of G, and indices $1,\ldots,t$ correspond to an arbitrary ordering of the vertices in the sparse graph candidate.

The high space connectivity requirement tends to select important examples and thus remove outliers and noisy points. Also, it is inclined to deemphasize the domination of groups of close points and maintain the manifold structure. This can be beneficial to many machine learning tasks, e.g., to classification problems. From the definition of (1), we can assume that points outside of the sparse graph G_s have high similarities to vertices in the graph, whereas high similarities of examples indicate that high similarities of labels can be expected. Consequently, should a good classifier be learned from the sparse graph, it tends to generalize well to unseen points with a high possibility.

2.1 Related work

A related concept called graph sparsification differs from our manifold-preserving sparse graphs both in motivation and technique. The task of graph sparsification is to approximate a graph by a sparse graph with the motivation of accelerating cut algorithms or solving linear equations in diagonally-dominant matrices [2,25]. The features of this generic sparse graphs include: (1) Sparse in the sense of the number of edges, not vertices; (2) Edge weights are usually different from those of the original graph; (3) Almost all sparse graphs are constructed by randomized techniques [2]. However, the manifold-preserving sparse graphs we propose here concern reducing the number of vertices rather than edges. The edge weights from the original graph to our sparse graphs need not change. Furthermore, the manifold-preserving graph reduction algorithm given in Section 3 is deterministic if the vertex with the maximum degree in a graph is unique.

There is also a family of methods on sparse low-rank approximations of general matrices which minimizes the Frobenius norm of the difference of the original matrix and the sparse matrix [20,24]. Nevertheless, these approaches have no considerations on manifold preservation, and thus address quite different issues with our method.

3 Manifold-preserving graph reduction algorithm

In this section, first we propose a greedy manifold-preserving graph reduction algorithm and show its performance on two case studies. Then, we give theoretical results on the manifold-preserving property and its approximation ratio.

3.1 Algorithm and case studies

Given a sparsity level or the number of retained vertices in sparse graphs, the problem of exactly seeking manifold-preserving sparse graphs is NP-hard. Here, we give a simple and efficient greedy algorithm to construct such sparse graphs. Due to its simplicity and high efficiency, applying this algorithm to large-scale data would be quite straightforward.

Define the degree $d(i)$ associated with vertex i to be $d(i) = \sum_{i \sim j} W_{ij}$ where $i \sim j$ means (i, j) are connected by an edge (if two vertices are not linked, their similarity is regarded as zero). Table 1 gives the pseudo code for the manifold-preserving graph reduction algorithm, where we seek t vertices from the original graph of m vertices. The manifold-preserving graph reduction algorithm first chooses the vertex with the maximum degree (if more than one vertex has the same maximum degree, we randomly pick one), and removes all the edges and weights linked to this vertex from the original graph. This step tends to select successive vertices with a high space connectivity. Then, it adds the selected vertex and associated edges and weights to the sparse graph G_s (which is null initially). The same procedure repeats on the resultant graphs until a fixed number of sought vertices is found. The sparse graph obtained only includes the selected vertices and edges linking these vertices, and the weights on the edges are directly taken from the original graph. A similar algorithm can be mimicked if the sparsity level rather than the number of vertices is adopted to build the sparse graph.

[Table 1 about here.]

Suppose t vertices are sought from an original graph with m vertices. Let d_E be the maximum number of edges linked to a vertex in the original graph. The computational complexity of our algorithm is less than

$$O\left[d_E\Big(m + (m-1) + \ldots + (m-t+1)\Big)\right] = O(d_E m t), \tag{2}$$

which is respectively linear with respect to d_E, m and t. Thus the manifold-preserving graph reduction algorithm is very efficient.

We illustrate the performance of our algorithm with random uniform sampling as a baseline on two synthetic graphs. The first graph, shown in Fig. 1, includes 66 vertices with identical weights on the edges, and the number of retained vertices (shown in red circles) in the sparse manifolds is fixed as 10. Clearly, the sparse graph found by the MP (manifold-preserving) graph reduction algorithm preserves a better manifold structure.

[Fig. 1 about here.]

The second case study uses the synthetic data shown in Fig. 4(b) from Section 6. Suppose we fix the sparsity level to be 50%. The selected examples by the MP graph reduction algorithm and random sampling for the same number of examples are given in Fig. 2, from which the superiority of our algorithm for manifold preserving is quite clearly evident.

[Fig. 2 about here.]

3.2 Analysis

We now proceed to give a bound on the expected connectivity of a randomly picked point outside of a sparse graph to the sparse graph. Before this theorem, we give a lemma on McDiarmid's inequality.

Lemma 1 (McDiarmid's inequality) *Let $X_1, \ldots, X_n$ be independent random variables taking values in a set A, and assume that $f : A^n \to \mathbb{R}$ satisfies*

$$\sup_{x_1,\ldots,x_n,\hat{x}_i \in A} |f(x_1, \ldots, x_n) - f(x_1, \ldots, \hat{x}_i, x_{i+1}, \ldots, x_n)| \leq c_i, \quad 1 \leq i \leq n.$$

Then for all $\varepsilon > 0$, we have

$$P\Big\{f(X_1, \ldots, X_n) - \mathbb{E}f(X_1, \ldots, X_n) \geq \varepsilon\Big\} \leq \exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^n c_i^2}\right).$$

Theorem 1 *Suppose we have a deterministic graph construction algorithm to build a graph $G(V, E, W)$ from a training set S with m i.i.d examples drawn from a distribution $\mathcal{D}$. Let t be the predetermined number of vertices for output sparse graphs. Let $\hat{\mathbb{E}}_{m-t}(G, G_t) = \sum_{i=t+1}^m (\max_{j=1,\ldots,t} W_{ij})$ be the empirical connectivity between an optimal sparse graph $G_t(V_t, E_t, W_t)$ and the $m-t$ vertices outside of the sparse graph. For all $\delta \in (0, 1]$, the expected connectivity $\mathbb{E}_{m-t}(G_t)$ with respect to the m vertices can be bounded, with probability at least $1 - \delta$ over random draws of samples of size m, as*

$$Pr_{\mathcal{S}\sim\mathcal{D}^m}\left(\mathbb{E}_{m-t}(G_t) \geq \hat{\mathbb{E}}_{m-t}(G, G_t) - (m-t)R\sqrt{\frac{1}{2}m\ln\frac{1}{\delta}}\right) \geq 1-\delta, \quad (3)$$

where R is the supremum of possible values of degree $d(i)$ for any vertex in a constructed graph. Then the expected connectivity $\mathbb{E}_{x\backslash V_t}(G_t)$ of a randomly picked point x outside of G_t to G_t, which equals $\frac{1}{m-t}\mathbb{E}_{m-t}(G_t)$, can be bounded with the same confidence as

$$Pr_{x\sim\mathcal{D}}\left(\mathbb{E}_{x\backslash V_t}(G_t) \geq \hat{\hat{\mathbb{E}}}_{m-t}(G, G_t) - R\sqrt{\frac{1}{2}m\ln\frac{1}{\delta}}\right) \geq 1-\delta, \quad (4)$$

where $\hat{\hat{\mathbb{E}}}_{m-t}(G, G_t) \triangleq \frac{1}{m-t}\hat{\mathbb{E}}_{m-t}(G, G_t)$ is the averaged empirical connectivity.

PROOF. Take f in Lemma 1 to be $\hat{\mathbb{E}}_{m-t}(G, G_t)$, which is clearly a function of examples $\{x_1, \ldots, x_m\}$ as a result of the application of a deterministic graph construction algorithm. It is simple to see that here c_i in Lemma 1 can be taken as $(m-t)R$. Note that the structure of the graph can change significantly even as a result of replacing one example. However, by the definition of R, we can still get the above estimate for c_i.

Given a confidence level we can apply Lemma 1 to the m vertices. For a specific G_t, setting the right hand side of the final inequality in Lemma 1 equal to δ results in $\varepsilon = c_i\sqrt{\frac{1}{2}m\ln\frac{1}{\delta}} = (m-t)R\sqrt{\frac{1}{2}m\ln\frac{1}{\delta}}$. Hence, we have

$$Pr_{S\sim\mathcal{D}^m}\left(\mathbb{E}_{m-t}(G_t) \leq \hat{\mathbb{E}}_{m-t}(G, G_t) - (m-t)R\sqrt{\frac{1}{2}m\ln\frac{1}{\delta}}\right) \leq \delta\ . \quad (5)$$

Negating this completes the proof of (3). The bound shown in (4) is directly obtained by dividing terms in the bracket of the left hand side of (3) by $m-t$. □

When confined to binary weights for the edges of the graphs, the expected connectivity $\mathbb{E}_{x\backslash V_t}(G_t)$ in this theorem is the expected number of edges a point outside of the sparse graph links to the most similar vertex in the sparse graph. Theorem 1 provides a guarantee of obtaining a good space connectivity. It indicates that for any point not in the sparse graph, its connectivity to the sparse manifold is greater than some value with a high probability. Maximizing (1) as required by Definition 3 can enlarge the lower bound provided in the theorem.

As directly seeking manifold-preserving sparse graphs is NP-hard, in this paper the algorithm in Table 1 is adopted to approximately maximize (1). It thus makes sense to characterize the approximation ratio, which is given below.

Theorem 2 *Given a graph $G(V, E, W)$ with m vertices, suppose G_s is the sparse graph obtained by the manifold-preserving graph reduction algorithm given in Table 1. Let d_{max} and w_{max} be the maximum degree and maximum weight of vertices in graph G, respectively. Define the connectivity between the sparse graph G_s and the remaining $m-t$ vertices as $C_{m-t}(G, G_s) = \sum_{i=t+1}^{m} (\max_{j=1,\dots,t} W_{ij})$. Then we can bound the approximation as*

$$\frac{C_{m-t}(G, G_s)}{\hat{\mathbb{E}}_{m-t}(G, G_t)} \geq \frac{d_{max} - C_{s,t}}{(m-t)w_{max}}, \tag{6}$$

where $\hat{\mathbb{E}}_{m-t}(G, G_t)$ was defined in Theorem 1, and $C_{s,t}$ is the sum of weights between the first selected vertex in G_s and the other $t-1$ vertices in G_s.

PROOF. By the procedure of the algorithm in Table 1, we know that the vertex with the maximum degree d_{max} is the first vertex selected to the sparse graph G_s. Denote this vertex as V_1. Therefore, the connectivity of the $m-t$ vertices out of G_s to V_1 is equal to $d_{max} - C_{s,t}$. According to its definition, $C_{m-t}(G, G_s)$ must be greater than or equal to this quantity. That is, $C_{m-t}(G, G_s) \geq d_{max} - C_{s,t}$.

On the other hand, we can bound $\hat{\mathbb{E}}_{m-t}(G, G_t)$ from above by $(m-t)w_{max}$, consulting the definition of w_{max}. Hence, the approximation ratio of our algorithm is bounded from below as

$$\frac{C_{m-t}(G, G_s)}{\hat{\mathbb{E}}_{m-t}(G, G_t)} \geq \frac{d_{max} - C_{s,t}}{(m-t)w_{max}}, \tag{7}$$

which is the required result by Theorem 2. □

Note that the lower bound given in Theorem 2 can be very loose in some cases. Take Fig. 3 as an example, where the sparse graph including four nodes is shown with a dashed circle. For this example, the numerator of the lower bound would be zero. As we do not know how to find a general tighter bound now, we leave it open for further research.

[Fig. 3 about here.]

We now attempt to compare the theoretical connectivities obtained by our algorithm in Table 1 and random sampling to assess the superiority of the proposed algorithm. As random sampling is a completely random algorithm, the selected sparse graph can be any one of the $\binom{m}{t}$ candidate sparse graphs. On a specific instantiation, the connectivity of the selected sparse graph can be the connectivity of any candidate sparse graph. Therefore, no guarantee is granted to choose a sparse graph with a high connectivity.

The expected connectivity of sparse graphs chosen by random sampling is the average of connectivities for all these candidate graphs. It is difficult to compare this quantity with $C_{m-t}(G, G_s)$ defined in Theorem 2. However, we can take a closer view of the performances of these two algorithms on two extreme situations. In the first case, if $t = 1$ which means that sparse graphs should only include one vertex, the sparse graph found by the proposed algorithm would be exactly the optimal sparse graph. But, random sampling will randomly pick a vertex from all m vertices, which is far from optimal. In the other case, suppose t in Table 1 is equal to t_{max}, where t_{max} is the number of iterations after which there are no edges left in the remaining graph. This is reminiscent of the natural vertex cover algorithm for the well-studied minimum vertex cover problem [12–14,17]. Now the sparse graph G_s has the property that every vertex outside of G_s has a common edge with some vertex in G_s. This indicates that the manifold would hardly change when G_s is used to replace the original full graph. However, with the random sampling method to choose t_{max} vertices, we cannot guarantee this appealing property. All the aforementioned analysis shows the advantage of the manifold-preserving graph reduction algorithm over random sampling.

In the remainder of this paper, we mainly consider the use of the sparse graphs for semi-supervised learning.

4 Sparse Laplacian SVMs

The method proposed here is based on Laplacian SVMs and regularization on sparse graphs. As a result of manifold-preserving graph reduction, the outliers and noisy examples in the training data tend to be eliminated, and our method will be fast in predicting the labels of new examples. As a related but different approach, the sparse Laplacian core vector machine [27] adopts the idea of sparsity in the decision rule. It adopts an error-insensitive regularizer to catch sparsity which can not screen out outliers and noisy examples.

4.1 Formulation

Here we show how the idea of using sparse graphs is elegantly combined with Laplacian SVMs to reach the sparse Laplacian SVMs.

By including a penalty term on the intrinsic manifold smoothness, Belkin et al. [6] introduced the Laplacian SVMs which solve the following problem

$$\min_{f \in \mathcal{H}} \frac{1}{l} \sum_{i=1}^{l} (1 - y_i f(x_i))_+ + \gamma_A \|f\|^2 + \frac{\gamma_I}{(l+u)^2} \mathbf{f}^\top L \mathbf{f}, \tag{8}$$

where $\mathcal{H}$ is the RKHS induced by a kernel, l and u are respectively the numbers of labeled and unlabeled examples, $(\cdot)_+ = \max\{0, \cdot\}$ is the hinge loss, γ_A and γ_I are respectively ambient and intrinsic regularization parameters, vector $\mathbf{f} = [f(x_1), \dots, f(x_{l+u})]^\top$, and L is the graph Laplacian.

For sparse Laplacian SVMs, we essentially have the following objective function

$$\min_{f, L} \frac{1}{l} \sum_{i=1}^{l} (1 - y_i f(x_i))_+ + \gamma_A \|f\|^2 + \frac{\gamma_I}{\|L\|_0^2} \mathbf{f}^\top L \mathbf{f} + \gamma_0 \|L\|_0, \tag{9}$$

where $\|L\|_0$ is the zero-norm of Laplacian matrix L and is related to the number of examples adopted for graph construction. For symmetric matrices as considered in the current context, $\|L\|_0$ is equal to the number of rows or columns whose entries are not all zeros. Regularization parameter γ_0 controls the importance of the sparsity, namely, the number of training examples used to construct the graph.

As optimizing the above problem with the norm $\|\cdot\|_0$ is NP-hard, we instead adopt the manifold-preserving graph reduction algorithm to find a sparse graph G_r whose Laplacian is L_r, and then perform optimization on this smaller graph. An alternative, which we will not try in this paper, is to replace the zero-norm of L in (9) by the one-norm which is known to lead to sparse solutions. Therefore, the objective of the sparse Laplacian SVMs becomes

$$\min_{f \in \mathcal{H}} \frac{1}{l} \sum_{i=1}^{l} (1 - y_i f(x_i))_+ + \gamma_A \|f\|^2 + \gamma_I \mathbf{f}^\top L_r \mathbf{f}, \tag{10}$$

where we have replaced $\frac{\gamma_I}{\|L_r\|_0^2}$ with γ_I. Since (10) seems quite similar to (8), this form of sparse Laplacian SVMs can be regarded as an application of the Laplacian SVMs to a certain set of unlabeled data. As labeled examples are usually few in semi-supervised learning settings, we include all the labeled examples in the sparse graph. In particular, we first construct a sparse graph based on the original graph from both labeled and unlabeled data, and then add the labeled examples to the sparse graph if they were removed during the graph reduction process. Suppose the size of the sparse graph is $l + r$ with $0 \leq r \leq u$. Then G_r includes $l+r$ vertices and the size of L_r is $(l+r) \times (l+r)$.

4.2 Optimization

By the representer theorem, the solution to (10) can be represented as $f(\cdot) = \sum_{j=1}^{l+r} \alpha_j k(x_j, \cdot)$, where k is the kernel function. The primal problem can be rewritten as:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\xi}} \;\; P_0 = \frac{1}{l}\sum_{i=1}^{l} \xi_i + \gamma_A \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} + \gamma_I \boldsymbol{\alpha}^\top K L_r K \boldsymbol{\alpha}$$
$$\text{s.t.} \begin{cases} y_i \left(\sum_{j=1}^{l+r} \alpha_j k(x_j, x_i)\right) \geq 1 - \xi_i, & i = 1, \ldots, l \\ \xi_i \qquad\qquad\qquad\qquad \geq 0, & i = 1, \ldots, l \; , \end{cases} \tag{11}$$

where $y_i \in \{-1, +1\}$, $\gamma_A \geq 0$, $\gamma_I \geq 0$, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{l+r}]^\top$, and K is the kernel matrix.

The Lagrangian $Lag(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\lambda}, \boldsymbol{\nu})$ can be written as

$$Lag = P_0 - \sum_{i=1}^{l} \left[\lambda_i \left(y_i \left(\sum_{j=1}^{l+r} \alpha_j k(x_j, x_i) \right) - 1 + \xi_i \right) + \nu_i \xi_i \right], \tag{12}$$

where $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_l]^\top \succeq 0$, $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_l]^\top \succeq 0$.

To obtain the Lagrangian dual function, function Lag should be minimized with respect to primal variables $\boldsymbol{\alpha}, \boldsymbol{\xi}$. For this purpose, we compute the corresponding partial derivatives and set them to 0, and thus obtain the following equalities

$$2(\gamma_A K + \gamma_I K L_r K)\boldsymbol{\alpha} = \sum_{i=1}^{l} \lambda_i y_i K(:, i), \tag{13}$$

$$\lambda_i + \nu_i = \frac{1}{l}, \tag{14}$$

where $K(:, i)$ denotes the ith column of K.

The Lagrangian is simplified as

$$
\begin{aligned}
Lag &= \gamma_A \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} + \gamma_I \boldsymbol{\alpha}^\top K L_r K \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \left(\sum_{i=1}^{l} \lambda_i y_i K(:,i) \right) + \sum_{i=1}^{l} \lambda_i \\
&= -\frac{1}{2} \boldsymbol{\alpha}^\top \left(\sum_{i=1}^{l} \lambda_i y_i K(:,i) \right) + \sum_{i=1}^{l} \lambda_i. \qquad (15)
\end{aligned}
$$

Denote $\gamma_A K + \gamma_I K L_r K$ by J, $\sum_{i=1}^{l} \lambda_i y_i K(:,i)$ by Λ. By (13) we have $2J\boldsymbol{\alpha} = \Lambda$. Thus $\boldsymbol{\alpha} = \frac{1}{2} J^{-1} \Lambda$. Now we see that the inverse of J involves a matrix sized $(l+r) \times (l+r)$ rather than the original $(l+u) \times (l+u)$. Therefore, sparse Laplacian SVMs can deal with large-scale data sets.

Thus, the Lagrange dual function $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is

$$
g = \inf_{\boldsymbol{\alpha}, \boldsymbol{\xi}} Lag = -\frac{1}{4} \Lambda^\top J^{-1} \Lambda + \sum_{i=1}^{l} \lambda_i. \qquad (16)
$$

Note that $\Lambda = K_l Y \boldsymbol{\lambda}$ with diagonal matrix $Y = diag(y_1, \ldots, y_l)$, and $K_l = K(:, 1:l)$.

Define $\tilde{A} = \frac{1}{2} Y K_l^\top J^{-1} K_l Y$ and $\mathbf{1} = (1, \ldots, 1_{(l)})^\top$. The Lagrange dual optimization problem can be formulated as

$$
\begin{aligned}
\min_{\boldsymbol{\lambda}} \quad & \frac{1}{2} \boldsymbol{\lambda}^\top \tilde{A} \boldsymbol{\lambda} - \mathbf{1}^\top \boldsymbol{\lambda} \\
\text{s.t.} \quad & 0 \preceq \boldsymbol{\lambda} \preceq \frac{1}{l} \mathbf{1}. \qquad (17)
\end{aligned}
$$

This convex optimization problem can be readily solved by standard software. Then we get $\boldsymbol{\alpha} = \frac{1}{2} J^{-1} K_l Y \boldsymbol{\lambda}$.

5 Empirical Rademacher complexity

The margin bound on the generalization performance for kernel-based classes developed in [21] (see Theorem 4.17 in page 102 with $\gamma = 1$) can be applied directly to the class of functions induced by sparse Laplacian SVMs corresponding to (10), which is literally slightly modified here as follows.

Theorem 3 *Fix $\delta \in (0,1)$ and let $\mathcal{F}$ be the class of functions induced by sparse Laplacian SVMs mapping from X to* $\boldsymbol{R}$*. Let $S = \{(x_1, y_1), \cdots, (x_l, y_l)\}$*

be drawn independently according to a probability distribution $\mathcal{D}$. Then with probability at least $1-\delta$ over samples of size l, every $f \in \mathcal{F}$ satisfies

$$P_{\mathcal{D}}\Big(y \neq sgn(f(\boldsymbol{x}))\Big) \leq \frac{1}{l}\sum_{i=1}^{l}\xi_i + 2\hat{R}_l(\mathcal{F}) + 3\sqrt{\frac{\ln(2/\delta)}{2l}},$$

where $\xi_i = (1 - y_i f(x_i))_+$, and $\hat{R}_l(\mathcal{F})$ is the empirical Rademacher complexity of $\mathcal{F}$

$$\hat{R}_l(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\left|\frac{2}{l}\sum_{i=1}^{l}\sigma_i f(x_i)\right|\Bigg| x_1,\ldots,x_l\right],$$

where $\boldsymbol{\sigma} = \{\sigma_1,\ldots,\sigma_l\}$ are independent uniform $\{\pm 1\}$-valued (Rademacher) random variables [1].

For $\hat{R}_l(\mathcal{F})$, we give the following Theorem 4. Note that this result also applies to the Laplacian SVMs with their corresponding labeled and unlabeled training examples. The technique used to prove Theorem 4 is given in the following of this section, which is analogical to that adopted in [18] for analyzing least-squares co-regularization.

Theorem 4 *Suppose $U^2 = tr[K_{l2}(\gamma_A K + \gamma_I K_r^\top L_{rr} K_r)^{-1} K_{l2}^\top]$ where K_{l2} and K_r are the first l and last r rows of K, respectively, and L_{rr} is the graph Laplacian of the graph only including r unlabeled examples. The empirical Rademacher complexity $\hat{R}_l(\mathcal{F})$ is bounded as $\frac{\sqrt{2}U}{l} \leq \hat{R}_l(\mathcal{F}) \leq \frac{2U}{l}$.*

Suppose $Q(f)$ is the objective function in (10). Plugging in the trivial predictors $f \equiv 0$ gives $\min_{f\in\mathcal{H}} Q(f) \leq Q(0) = 1$.

We have

$$\begin{aligned}\mathbf{f}^\top L_r \mathbf{f} &= \frac{1}{2}\sum_{i,j=1}^{l+r} W_{ij}(f(x_i) - f(x_j))^2 \\ &\geq \frac{1}{2}\sum_{i,j=l+1}^{l+r} W_{ij}(f(x_i) - f(x_j))^2 = \mathbf{f}_r^\top L_{rr}\mathbf{f}_r\end{aligned}$$

where $\mathbf{f}_r = [f(x_{l+1}),\ldots,f(x_{l+r})]^\top$, and L_{rr} is the graph Laplacian of the sparse graph including only the r unlabeled examples. Since all terms in $Q(f)$ are nonnegative, we conclude that any f^* minimizing $Q(f)$ is contained in $\tilde{\mathcal{H}} = \{f : \gamma_A\|f\|^2 + \gamma_I {\mathbf{f}_r}^\top L_{rr}\mathbf{f}_r \leq 1\}$.

Therefore, the complexity $\hat{R}_l(\mathcal{F})$ is

$$\hat{R}_l(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{H}}\left|\frac{2}{l}\sum_{i=1}^{l}\sigma_i f(x_i)\right| : f \in \tilde{\mathcal{H}}\right], \tag{18}$$

which can be bounded as shown in the following two subsections.

5.1 Supremum conversion

Since $f \in \tilde{\mathcal{H}}$ implies $-f \in \tilde{\mathcal{H}}$, we can drop the absolute sign in (18). Define $\mathcal{L} = \text{span}\{k(\mathbf{x}_i, \cdot)\}_{i=1}^{l+r} \in \mathcal{H}$. We know that the supremum in $\hat{R}_l(\mathcal{F})$ is unchanged if we restrict the domain of the supremum to $f \in \mathcal{L} \cap \tilde{\mathcal{H}}$.

The class of functions $\mathcal{L} \cap \tilde{\mathcal{H}}$ can be formulated as

$$\begin{aligned} \mathcal{L} \cap \tilde{\mathcal{H}} &= \{f_{\boldsymbol{\alpha}} : \gamma_A \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} + \gamma_I \boldsymbol{\alpha}^\top K_r^\top L_{rr} K_r \boldsymbol{\alpha} \leq 1\} \\ &= \{f_{\boldsymbol{\alpha}} : \boldsymbol{\alpha}^\top N \boldsymbol{\alpha} \leq 1\}, \end{aligned} \tag{19}$$

where $K_r = K(l+1 : l+r, :)$ is the last r rows of K, and $N = \gamma_A K + \gamma_I K_r^\top L_{rr} K_r$. Now we can write $\hat{R}_l(\mathcal{F}) = \frac{2}{l} \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\boldsymbol{\alpha} \in \mathcal{R}^{l+r}} \{\boldsymbol{\sigma}^\top K_{l2} \boldsymbol{\alpha} : \boldsymbol{\alpha}^\top N \boldsymbol{\alpha} \leq 1\}$ with $K_{l2} = K(1 : l, :)$ being the first l rows of K.

For a symmetric positive definite matrix Θ, it is simple to show that [18]

$$\sup_{\boldsymbol{\alpha} : \boldsymbol{\alpha}^\top \Theta \boldsymbol{\alpha} \leq 1} \mathbf{v}^\top \boldsymbol{\alpha} = \|\Theta^{-1/2} \mathbf{v}\|.$$

Without loss of generality, suppose positive semi-definite matrix N is positive definite. Thus, we can evaluate the supremum as described above to get

$$\hat{R}_l(\mathcal{F}) = \frac{2}{l} \mathbb{E}_{\boldsymbol{\sigma}} \|N^{-1/2} K_{l2}^\top \boldsymbol{\sigma}\|.$$

5.2 Bounding $\hat{R}_l(\mathcal{F})$

Lemma 2 (Kahane-Khintchine inequality [16]) *For any vectors $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n$ in a Hilbert space and independent Rademacher random variables $\sigma_1, \cdots, \sigma_n$, the following holds*

$$\frac{1}{2} \mathbb{E} \| \sum_{i=1}^{n} \sigma_i \boldsymbol{a}_i \|^2 \leq \left(\mathbb{E} \| \sum_{i=1}^{n} \sigma_i \boldsymbol{a}_i \| \right)^2 \leq \mathbb{E} \| \sum_{i=1}^{n} \sigma_i \boldsymbol{a}_i \|^2.$$

By Lemma 2 we have $\frac{\sqrt{2}U}{l} = \frac{2U}{\sqrt{2}l} \leq \hat{R}_l(\mathcal{F}) \leq \frac{2U}{l}$ where

$$\begin{aligned} U^2 &= \mathbb{E}_{\boldsymbol{\sigma}} \|N^{-1/2} K_{l2}^\top \boldsymbol{\sigma}\|^2 \\ &= tr \left[K_{l2} (\gamma_A K + \gamma_I K_r^\top L_{rr} K_r)^{-1} K_{l2}^\top \right], \end{aligned} \tag{20}$$

from which we see that the roles of γ_A and γ_I are parallel, taking effect only through the term $\gamma_A K + \gamma_I K_r^\top L_{rr} K_r$. The proof of Theorem 4 is now completed.

5.3 Discussion

From Theorem 3 and Theorem 4, we can relate the quantity U to the generalization bound, and also characterize the potential impact of other quantities. The expression of U shows that the two regularization terms in (10) related to γ_A and γ_I play parallel roles in the generalization bound. Moreover, these roles have different dependencies on matrices K and L_{rr} as reflected by (20).

In addition, U is a nonlinear function of K and L_{rr}. Although it can be hard to determine the exact ranges of K and L_{rr} for which U would be monotonically increasing or decreasing, the expression of U indeed raises the possibility of motivating new models, e.g., using U as a regularization term in the objective function. We does not investigate this issue further in this paper.

An open problem is concerned with the possibility of relating U to the connectivity given in Definition 3. For instance, if high connectivities correspond to low values of U, the rigorous theoretical justification of Definition 3 for classification would be found in terms of generalization bounds.

5.4 Related work

There are some other theoretical works on graph-based semi-supervised learning in the literature. Here we briefly review these works to provide readers a better understanding of this field.

Zhang and Ando [28] and Johnson and Zhang [11] gave a generalization bound for graph-based transductive learning, which can be used to analyze the effect of different kernels. Although different from our theoretical result, the complexity term in their bound also relies on the trace of some related matrices. Belkin et al. [3] derived a bound on the generalization error of transductive learning using the notion of algorithmic stability, where the smallest nontrivial eigenvalue of the smoothness matrix (e.g., the graph Laplacian) plays an important role. This result indicates that a larger value of this eigenvalue can lead to a lower error bound while the size of the graph is relatively unimportant. Johnson and Zhang [10] investigated the theoretical effect of Laplacian normalization in multi-class transductive learning on graphs, which reveals the limitations of the standard degree-based normalization method. They further proposed a remedy to overcome the limitations. El-Yaniv and Pechyony [8] defined the transductive Rademacher complexity based on which they derived a generalization bound for transductive learning. For specific algorithms, this complexity can be bounded from the unlabeled-labeled data decomposition which applies to many graph-based algorithms [8,9].

Some other related work from the perspective of reducing noise from graphs include [4] and [5]. They used the Laplace-Beltrami operator to produce an eigenfunction basis, and then conducted learning in the submanifold constituted by some leading eigenfunctions. This kind of noise reduction is implemented in terms of dimension reduction, while we carry out noise reduction by removing examples in this paper.

6 Experiments

In this section, we report experimental results of the sparse Laplacian SVMs on four data sets. For graph adjacency construction, the k-nearest-neighbor rule is used where k is set to 10. For graph weight computation on the first two data sets, the Gaussian RBF (radial basis function) kernel is used

$$W_{ij} = \begin{cases} \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2}), & x_i, x_j \text{ are neighbors}, \\ 0, & \text{otherwise} . \end{cases} \tag{21}$$

On the other two data sets, the polynomial kernel is adopted

$$W_{ij} = \begin{cases} (x_i^\top x_j)^p, & x_i, x_j \text{ are neighbors}, \\ 0, & \text{otherwise} . \end{cases} \tag{22}$$

The kernels used for graph weight computation also serve as the kernel function for the classifiers of sparse Laplacian SVMs. For the manifold-preserving graph reduction algorithm, we fix the number of unlabeled examples retained in sparse manifolds with a proportion from 0.1 to 1. The baseline method is choosing at random the same number of unlabeled examples, since there are no other methods proposed so far for manifold-preserving graph reduction.

For each data set, given the number of labeled examples we choose them at random for 20 times, and report the averaged accuracies (for clarity, the standard deviations are omitted as the manifold-preserving graph reduction algorithm only gets slightly lower variances than the random sampling algorithms in our experiments) on test sets. The regularization parameters in sparse Laplacian SVMs are selected from the set $\{10^{-6}, 10^{-4}, 10^{-2}, 1, 10, 100\}$ by validation sets.

For comparison, we also report the performance of supervised SVMs under the same setting.

6.1 Noise-free synthetic data

[Fig. 4 about here.]

[Fig. 5 about here.]

This data set includes a training set of 200 examples (shown in Fig. 4(a)), a validation set of 100 examples, and a test set of 100 examples. For graph weight computation and sparse Laplacian SVMs, an RBF kernel with kernel parameter $\sigma = 0.35$ is used following the setting in [6]. The number of labeled examples in the training set is fixed as four (two from each class); the other examples in the training set are treated as unlabeled. Note that in this paper we use many more labeled examples in validation sets than in training sets, since model selection with few labeled examples for semi-supervised learning is still an open problem.

Fig. 5 shows the classification accuracies of sparse Laplacian SVMs with the manifold-preserving graph reduction (MPGR) algorithm and random sampling (Ran), and supervised SVMs. We see that the manifold-preserving graph reduction algorithm outperforms the other algorithms, and as expected for noise-free data more unlabeled examples retained in the graph would help.

We also perform another experiment different to that shown in Fig. 5, which uses the validation set to choose the best proportions of unlabeled examples retained where if two proportions hit the same accuracy the smaller proportion would be chosen. The result shows that the average value of the best proportions is 21.00% with an averaged test accuracy 99.65%.

6.2 Noisy synthetic data

[Fig. 6 about here.]

This data set differs to the noise-free data only in the addition of Gaussian white noise (shown in Fig. 4(b)). The experimental result is given in Fig. 6 where the same experimental setting is adopted. We get a similar conclusion with the noise-free data: basically the MPGR algorithm leads to the best performance. We also get a new insight from Fig. 6: keeping as many unlabeled examples in the noisy training set is not optimal. Our algorithm exhibits its effectiveness on the removal of outliers and noisy points.

Using the validation set to select the best proportions of unlabeled examples retained in sparse graphs shows that the averaged best proportion is 33.00% with a test accuracy 92.00%.

6.3 USPS data

Digits 3 and 8 from the USPS digital image data set are used. The training set and validation set include 700 and 100 digits, respectively, with halves of them being digit 3. The test set includes 424 digits of 3 and 308 digits of 8. Two examples from each class of the training set are randomly chosen as labeled ones, and the other examples are regarded as unlabeled. The training, validation and test sets are randomly selected for each of the 20 runs.

[Fig. 7 about here.]

For graph weight computation and sparse Laplacian SVMs, we use a polynomial kernel with degree $p = 3$ as in [6] for digit recognition. The classification results of different algorithms are given in Fig. 7, which indicates the superiority of our graph reduction algorithm. In addition, the averaged test accuracy provided by the best proportions of unlabeled examples is 93.83% which corresponds to the averaged proportion of 36.50%. The advantage of semi-supervised learning using partial unlabeled data is again justified.

6.4 MNIST data

Here a larger data set is used to evaluate algorithms, which includes digits 3 and 8 randomly selected from the MNIST digital image data. The training set and validation sets include 1900 and 100 digits, respectively, where digits 3 and 8 have an equal proportion. The test set consists of 1984 digits with 1010 of them being digit 3. The experimental setting including the number of labeled training examples and kernel functions is identical to that for the USPS data.

[Fig. 8 about here.]

Fig. 8 shows the classification accuracies of different algorithms. Clearly, MPGR gives the best performance, which outperforms the other methods greatly especially when the number of retained unlabeled examples is small. Experiments also show that the averaged best proportion of unlabeled examples retained is 28.00%, which corresponds to the averaged test accuracy 91.00%.

7 Conclusion

In this paper, we proposed manifold-preserving sparse graphs and a simple and efficient graph reduction algorithm to construct the sparse graphs. Applying them to Laplacian SVMs, we further proposed sparse Laplacian SVMs for semi-supervised learning. The main theoretical contributions of this paper are: presenting a theorem on the expected connectivity of a randomly picked point outside of a sparse graph to the sparse graph; bounding the approximation ratio of the proposed graph reduction algorithm; giving the empirical Rademacher complexity of the function class induced by the sparse Laplacian SVMs.

Experimental results on multiple data sets have shown that the sparse Laplacian SVMs using the manifold-preserving sparse graphs outperform those based on random sampling, and get a good sparsity level (a large portion of the examples can be removed without sacrificing the accuracy much). Especially when training data include outliers and noisy examples, the manifold-preserving graph reduction algorithm can effectively remove them and even improve performance.

A general trend from the graphs in experiments is that the classification accuracy often first increases and then decreases as the proportion of unlabeled examples retained increases. We conjecture that this is due to the fact that when a small number of unlabeled examples are retained, the manifold structures can be well approximated, and when a large number of them are kept, there would be a large possibility that examples from different categories gather and overlap and thus the manifold structures are damaged. These examples leading to a bad manifold representation act like outliers and noisy examples to some extent. Our proposed manifold-preserving graph reduction algorithm has the capability to select partial data that are important to represent manifold structures and thus obtain good classification performance.

Application of the manifold-preserving sparse graphs to other learning contexts would be interesting future work. For example, by manually labeling the unlabeled examples retained in sparse graphs, it is likely to get a promising active learning method.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Project 61075005, the Fundamental Research Funds for the Central Universities, and the PASCAL network of excellence.

References

[1] P. Bartlett, S. Mendelson, Rademacher and Gaussian complexities: Risk bounds and structural results, Journal of Machine Learning Research 3 (2002) 463–482.

[2] J. Batson, D. Spielman, N. Srivastava, Twice-Ramanujan sparsifiers, in: Proceedings of the 41st Annual ACM Symposium on the Theory of Computing, 2009, pp. 255–262.

[3] M. Belkin, I. Matveeva, P. Niyogi, Regularization and semi-supervised learning on large graphs, Lecture Notes in Computer Science 3120 (2004) 624–638.

[4] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Computation 15 (2003) 1373–1396.

[5] M. Belkin, P. Niyogi, Semi-supervised learning on Riemannian manifolds, Machine Learning 56 (2004) 209–239.

[6] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, Journal of Machine Learning Research 7 (2006) 2399–2434.

[7] O. Chapelle, B. Schökopf, A. Zien, Semi-supervised Learning, MIT Press, Cambridge, MA, 2006.

[8] R. El-Yaniv, D. Pechyony, Transductive Rademacher complexity and its applications, Lecture Notes in Computer Science 4539 (2007) 157–171.

[9] R. El-Yaniv, D. Pechyony, V. Vapnik, Large margin vs. large volume in transductive learning, Machine Learning 72 (2008) 173–188.

[10] R. Johnson, T. Zhang, On the effectiveness of Laplacian normalization for graph semi-supervised learning, Journal of Machine Learning Research 8 (2007) 1489–1517.

[11] R. Johnson, T. Zhang, Graph-based semi-supervised learning and spectral kernel design, IEEE Transactions on Information Theory 54 (2008) 275–288.

[12] G. Karakostas, A better approximation ratio for the vertex cover problem, ACM Transactions on Algorithms 5 (2009) Article 41.

[13] R. Karp, R. Miller, J. Thatcher, Reducibility among combinatorial problems, The Journal of Symbolic Logic 40 (1975) 618–619.

[14] S. Khot, O. Regev, Vertex cover might be hard to approximate to within $2-\epsilon$, Journal of Computer and System Sciences 74 (2008) 335–349.

[15] G. Kimeldorf, G. Wahba, Some results on Tchebycheffian spline functions, Journal of Mathematical Analysis and Applications 33 (1971) 82–95.

[16] R. Latala, K. Oleszkiewicz, On the best constant in the Khintchine-Kahane inequality, Studia Mathematica 109 (1994) 101–104.

[17] C. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes, Journal of Computer and System Sciences 43 (1991) 425–440.

[18] D. Rosenberg, P. Bartlett, The Rademacher complexity of co-regularized kernel classes, Journal of Machine Learning Research Workshop and Conference Proceedings 2 (2007) 396–403.

[19] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.

[20] M. Rudelson, R. Vershynin, Sampling from large matrices: An approach through geometric functional analysis, Journal of the ACM 54 (2007) Article 21.

[21] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, Cambridge, England, 2004.

[22] V. Sindhwani, P. Niyogi, M. Belkin, A co-regularization approach to semi-supervised learning with multiple views, in: Proceedings of the 22nd ICML Workshop on Learning with Multiple Views, 2005.

[23] V. Sindhwani, D. Rosenberg, An RKHS for multi-view learning and manifold co-regularization, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 976–983.

[24] A. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in: Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 911–918.

[25] D. Spielman, N. Srivastava, Graph sparsification by effective resistances, in: Proceedings of the 40th Annual ACM Symposium on the Theory of Computing, 2008, pp. 563–568.

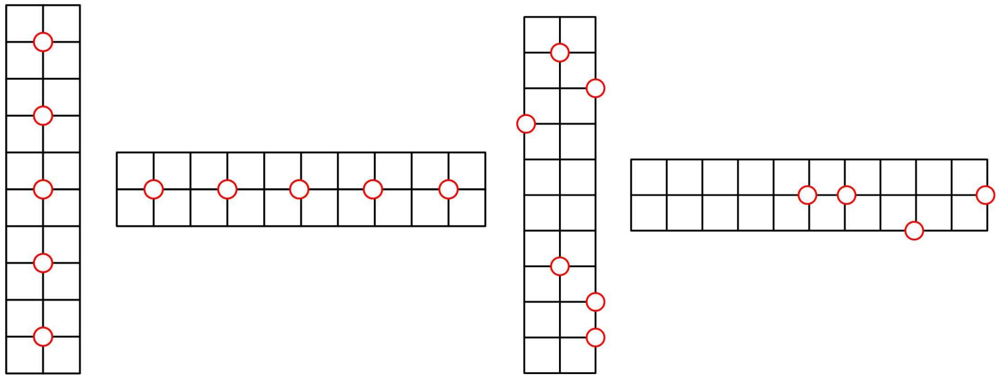
[26] J. Tenenbaum, V. de Silva, J. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.

[27] I. Tsang, J. Kwok, Large-scale sparsified manifold regularization, Advances in Neural Information Processing Systems 19 (2007) 1401–1408.

[28] T. Zhang, R. Ando, Analysis of spectral kernel design based semi-supervised learning, Advances in Neural Information Processing Systems 18 (2006) 1601–1608.

[29] X. Zhu, Semi-supervised learning literature survey, Technical Report 1530, University of Wisconsin-Madison, 2008.

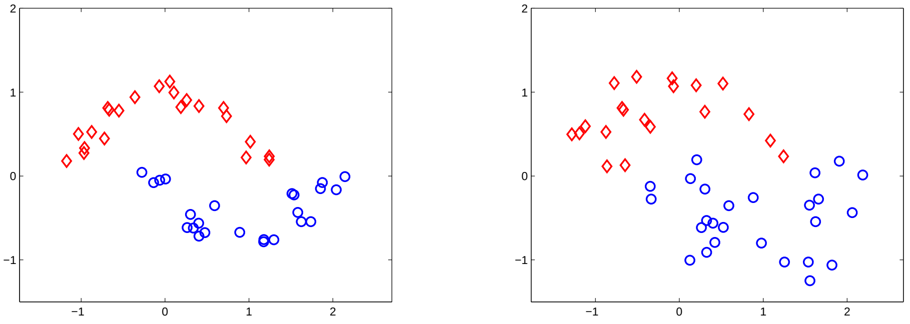
List of Figures



(a) MP graph reduction

(b) Random sampling

Fig. 1. Points retained in sparse manifolds with different methods.



(a) MP graph reduction

(b) Random sampling

Fig. 2. Points retained in sparse manifolds with different methods.

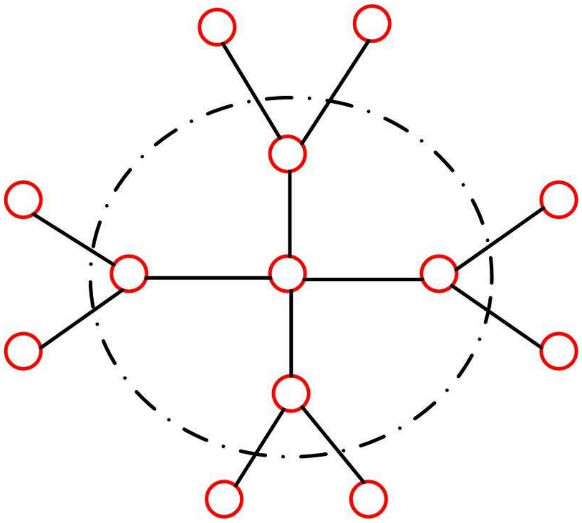
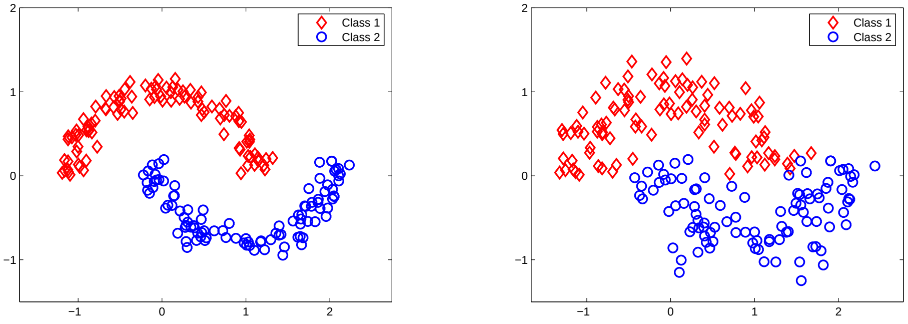


Fig. 3. A graph and its sparse graph with four nodes.



(a) Noise-free synthetic data

(b) Noisy synthetic data

Fig. 4. Training examples of the synthetic data sets.

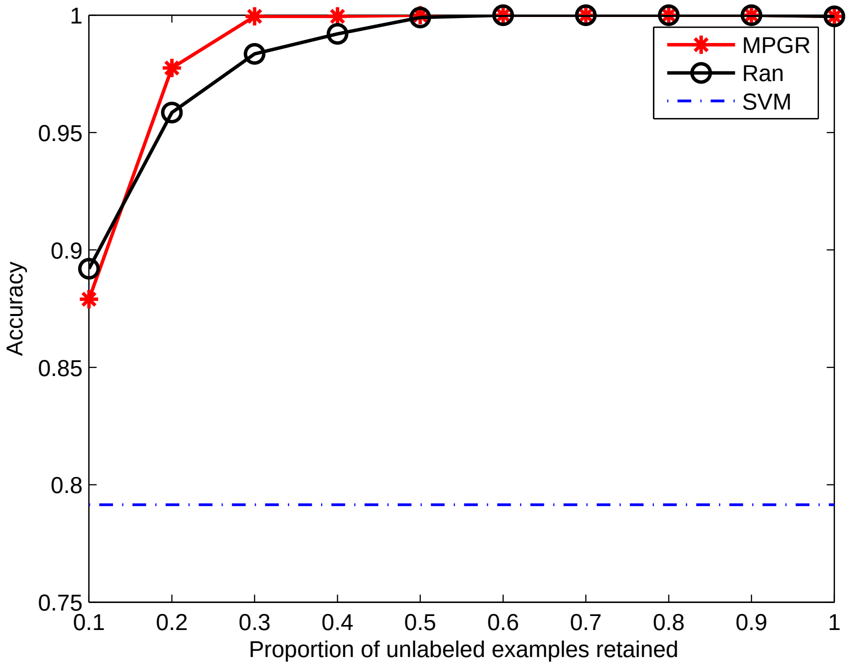


Fig. 5. Classification performance of different algorithms on the noise-free data.

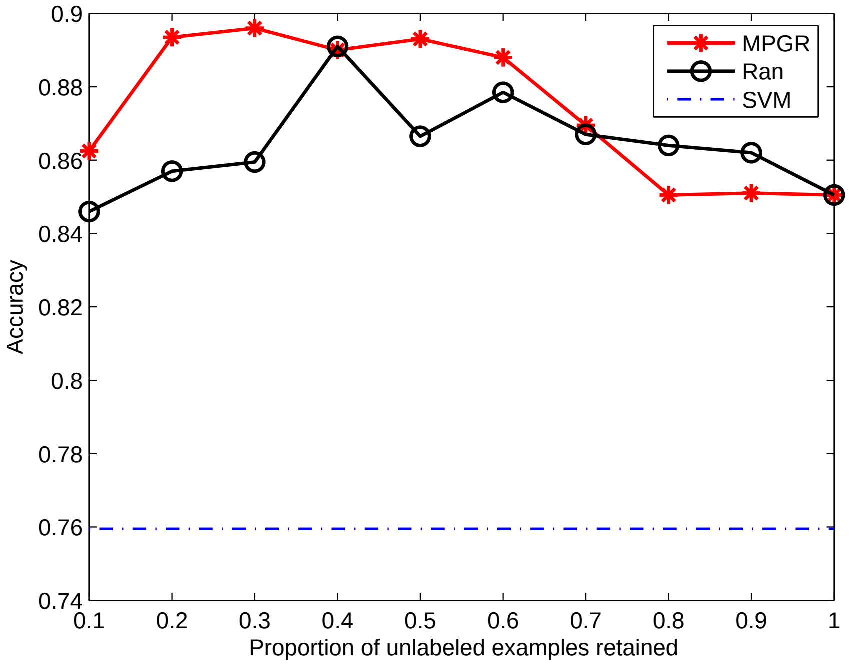


Fig. 6. Classification performance of different algorithms on the noisy data.

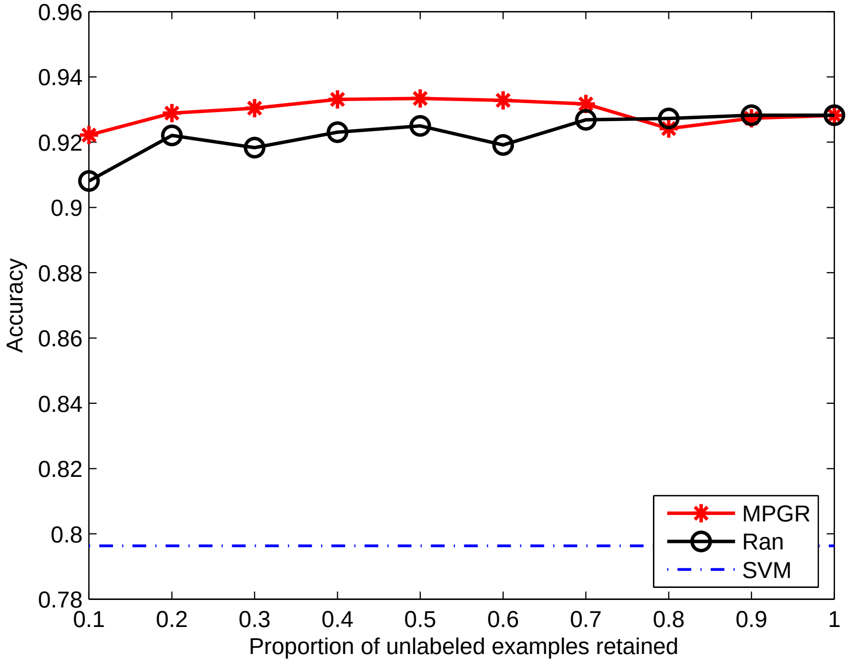


Fig. 7. Classification performance of different algorithms on the USPS data.

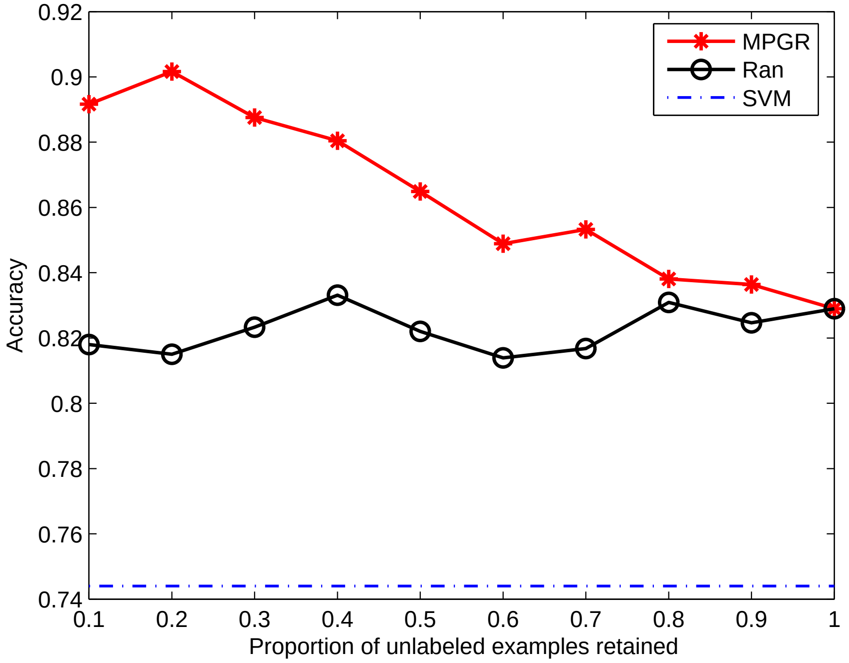


Fig. 8. Classification performance of different algorithms on the MNIST data.

List of Tables

Table 1
Manifold-Preserving Graph Reduction Algorithm

```
Input: Graph $G(V, E, W)$ with $m$ vertices;
       $t$ for the number of the vertices in the desired sparse graph $G_s$.
for $j = 1, \ldots, t$
    compute degree $d(i)$ $(i = 1, \ldots, m - j + 1)$
    pick one vertex $v$ with the maximum degree
    remove $v$ and associated edges from $G$; add $v$ to $G_s$
end for
Output: Manifold-preserving sparse graph $G_s$ with $t$ vertices.
```