BoostCluster: Boosting Clustering by Pairwise Constraints

Yi Liu, Rong Jin, and Anil K. Jain
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824, USA
{liuyi3, rongjin, jain}@cse.msu.edu

ABSTRACT

Data clustering is an important task in many disciplines. A large number of studies have attempted to improve clustering by using the side information that is often encoded as pairwise constraints. However, these studies focus on designing special clustering algorithms that can effectively exploit the pairwise constraints. We present a boosting framework for data clustering, termed as **BoostCluster**, that is able to iteratively improve the accuracy of *any* given clustering algorithm by exploiting the pairwise constraints. The key challenge in designing a boosting framework for data clustering is how to influence an arbitrary clustering algorithm with the side information since clustering algorithms by definition are unsupervised. The proposed framework addresses this problem by dynamically generating new data representations at each iteration that are, on the one hand, adapted to the clustering results at previous iterations by the given algorithm, and on the other hand consistent with the given side information. Our empirical study shows that the proposed boosting framework is effective in improving the performance of a number of popular clustering algorithms (K-means, partitional SingleLink, spectral clustering), and its performance is comparable to the state-of-the-art algorithms for data clustering with side information.

Categories and Subject Descriptors

I.5.3 [**Clustering**]: Algorithms; H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms

Algorithms

Keywords

Boosting, Data clustering, Semi-supervised learning, Pairwise constraints

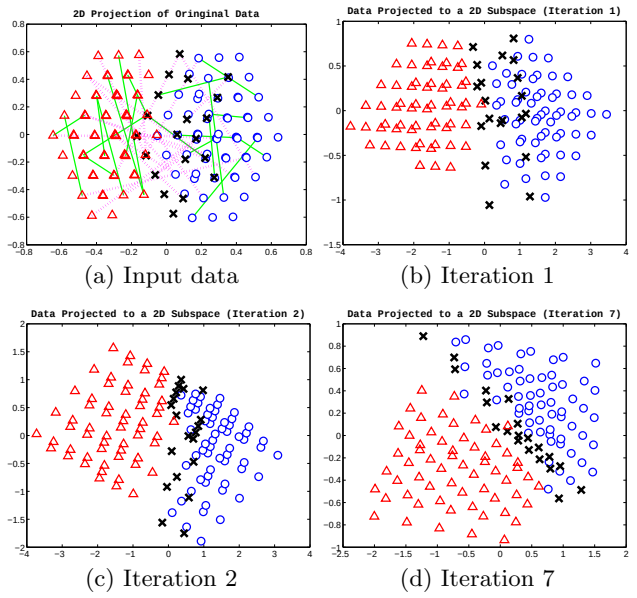
KDD'07, August 12–15, 2007, San Jose, California, USA.

1. INTRODUCTION

Data clustering, also called *unsupervised learning*, is one of the key techniques in data mining that is used to understand and mine the structure of unlabeled data. The idea of improving clustering by side information, sometimes called *semi-supervised clustering* or *constrained data clustering*, has received significant amount of attention in recent studies on data clustering. Often, the side information is presented in the form of pairwise constraints: the *must-link* pairs where data points should belong to the same cluster, and the *cannot-link* pairs where data points should belong to different clusters. There are two major approaches to semi-supervised clustering: the constraint-based approach and the approach based on distance metric learning. The first approach employs the side information to restrict the solution space, and only finds the solution that is consistent with the pairwise constraints. The second approach first learns a distance metric from the given pairwise constraints, and computes the pairwise similarity using the learned distance metric. The computed similarity matrix is then used for data clustering.

Although a large number of studies have been devoted to semi-supervised clustering, most of them focus on designing special clustering algorithms that can effectively exploit the pairwise constraints. For instance, algorithms in [4, 5, 23] are designed to improve the probabilistic models for data clustering by incorporating the pairwise constraints into the priors of the probabilistic models; the constrained K-means algorithm [27] exploits the pairwise constraints by adjusting the cluster memberships to be consistent with the given constraints. However, it is often the case that we have a specific clustering algorithm that is specially designed for the target domain, and we are interested in improving the accuracy of this clustering algorithm by the available side information. To this end, we propose a boosting framework for data clustering, termed as **BoostCluster**, that is able to improve *any* given clustering algorithm by the pairwise constraints. It is important to note that the proposed algorithm does not make any assumption about the underlying clustering algorithm, and is therefore applicable to any clustering algorithm.

Although a number of boosting algorithms (e.g., [10]) have been successfully applied to supervised learning, extending boosting algorithms to data clustering is significantly more challenging. The key difficulty is how to influence an arbitrary clustering algorithm with the given pairwise constraints. To overcome this challenge, we propose to encode the side information into the data representation that is the



(a) Input data (b) Iteration 1

(c) Iteration 2 (d) Iteration 7

Figure 1: An example illustrating the idea of iterative data projections. (a) shows the original data distribution, projected to the space spanned by its two principal components; (b)-(d) show the data distributions based on the new representations in the projected subspaces that are generated in iteration 1, 2, and 7.

input to the clustering algorithm. More specifically, we will first find the subspace in which data points of the must-link pairs are close to each other while data points of the cannot-link pairs are far apart. Then, a new data representation is generated by projecting all the data points into the subspace, and will be used by the given clustering algorithm to find the appropriate cluster assignments. Furthermore, the procedure for identifying the appropriate subspace based on the remaining unsatisfied constraints, and the procedure for clustering data points using the newly generated data representation will alternate iteratively till most of the constraints are satisfied. Although the idea of incorporating constraints into clustering through the generation of new data representations is not completely new, the existing approaches [12, 17, 19, 29] do not take into account the performance of the clustering algorithms while generating the data representations, and therefore only achieve sub-optimal performance.

Figure 1 illustrates the idea of iterative data projection. The data points used in this illustration are sampled from the "scale" dataset that will be described later in Section 4.1. They belong to three clusters that are labeled in Figure 1 by legends $\triangle$, $\circ$, and $\times$, respectively. A partitional clustering algorithm is used in this illustration. Sub-figure (a) shows the original data distribution projected into a 2D space that is generated by PCA. We clearly see that many data points of the cluster $\times$ overlap heavily with the data points of the clusters $\triangle$ and $\circ$, and they are difficult to be separated. The must-link and cannot-link constraints are indicated in Figure 1(a) by solid lines and dotted lines, respectively. Figure 1(b)-(d) illustrate the projected data distributions based on the new representations that are generated by the proposed boosting framework in iteration 1, 2, and 7, respectively. The data representations generated in different iterations are helpful in separating the data points in the cluster $\times$ from those in the other two clusters.

The remaining paper is arranged as follows: Section 2 briefly reviews the previous work on semi-supervised clustering. Section 3 describes the problem of boosting a given clustering algorithm by a set of pairwise constraints and introduces the proposed BoostCluster framework. Section 4 presents the results of our empirical study. Section 5 states conclusions of this work.

2. RELATED WORK

The constraint-based approach for semi-supervised clustering utilizes the side information to restrict the feasible solutions when deciding the cluster assignment. Early work in this category took the side information as hard constraints, and only considered the cluster assignments that were absolutely consistent with the given pairwise constraints. In [6, 27], the authors proposed the constrained K-means algorithm by adjusting the cluster memberships to be consistent with the pairwise constraints. In [26], a generalized Expectation Maximization (EM) algorithm is proposed to incorporate the pairwise constraints into the EM algorithm. In particular, the cluster assignments that are inconsistent with the constraints are excluded from the partition function when computing the posterior probability for the cluster memberships. One problem with treating the side information as hard constraints is that we may not be able to find feasible solutions that are consistent with all the constraints [7]. To overcome this problem, a number of studies view the side information as *soft* constraints. The key idea is to penalize, not to exclude, the cluster assignments that are inconsistent with the given pairwise constraints. In [4,5,23], the authors present probabilistic models for semi-supervised clustering where the pairwise constraints are incorporated into the clustering algorithms through the Bayesian priors. In [22], the authors modified the mixture model for data clustering by redefining the data generation process through the introduction of hidden variables. In [3], the authors extend the framework of semi-supervised clustering by selecting the most informative pairwise constraints to solicit the labeling information. In [8], the authors study semi-supervised clustering for the hierarchical clustering algorithm. In [21], a mean field approximation method was proposed to learn from constraint data. In [17], a spectral learning framework was proposed to incorporate the side information into data clustering.

Another approach to semi-supervised clustering is to first learn a distance metric from the given pairwise constraints. The pairwise similarity between any two examples is then computed based on the learned distance metric, and a clustering algorithm is applied to the computed similarity matrix. The key to this approach is to effectively learn a distance metric from the side information. Zhang et al. [32] proposed to learn a distance metric by a linear regression model. Xing et al. [29] formulated the distance metric learning problem as a constrained convex programming problem. This algorithm is extended to the nonlinear case in [20] by the introduction of kernels. Yang et al. [30] proposed a local distance metric algorithm that is designed to address the problem of distance metric learning for multi-modal data distributions. Golderberg et al. [11] presented the neighborhood compo-

nent analysis algorithm that learns a local distance metric by extending the nearest neighbor classifier. Weinberger [28] presented a large margin nearest-neighbor classifier for distance metric learning that extended the neighborhood component analysis to a maximum margin framework. Discriminative component analysis [15] learns a distance metric by extending the relevance component analysis to effectively explore both the must-link and the cannot-link constraints simultaneously. In [13,14], the authors extended the boosting algorithms to learn a distance function from given pairwise constraints. Schultz and Joachims [25] extended the framework of support vector machine to learn distance metrics from the pairwise comparisons.

Finally, a few studies cluster data points by a similarity matrix that is directly modified according to the pairwise constraints. In [19], the authors proposed to modify the similarity matrix by linearly combining the original similarity matrix with the pairwise constraints. Klein et al. [18] proposed to modify the similarity matrix by propagating the pairwise constraints through the nearest neighbors.

3. BOOSTING CLUSTERING

Let $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ denote the collection of examples to be clustered, where n is the total number of examples and each example $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of d dimensions. We use a matrix $S^+ \in \mathbb{R}^{n \times n}$ to represent all the must-link pairs, where $S^+_{i,j}$ is one when examples $\mathbf{x}_i$ and $\mathbf{x}_j$ form a must-link pair, and zero otherwise. Similarly, we use a matrix $S^- \in \mathbb{R}^{n \times n}$ to represent all the cannot-link pairs, where $S^-_{i,j}$ is one when examples $\mathbf{x}_i$ and $\mathbf{x}_j$ form a cannot-link pair, and zero otherwise. Let $\mathcal{A}$ denote the given clustering algorithm to be improved. In order to make this framework general, we treat the clustering algorithm $\mathcal{A}$ as a black box that only takes the data representation of all examples as its input and outputs the cluster memberships for the given examples. Note in this work, we assume that the number of clusters is given.

3.1 Objective Function

The first step toward the boosting algorithm is to construct an appropriate objective function. As described in the introduction section, the goal of the boosting algorithm is to identify the subspace that keeps the data points in the unsatisfied must-link pairs close to each other, and keeps the data points from the unsatisfied cannot-link pairs well separated. In order to identify which constraint pairs are not well satisfied, we introduce the kernel similarity matrix $K \in \mathbb{R}^{n \times n}$, where $K_{i,j} \geq 0$ indicates the confidence of assigning examples $\mathbf{x}_i$ and $\mathbf{x}_j$ to the same cluster. When all the constraints are satisifed, we expect to observe a large value for kernel similarity $K_{i,j}$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ form a must-link pair, and a small value for $K_{i,j}$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ form a cannot-link pair. Hence, we propose the following objective function to measure the inconsistency between the kernel matrix K and the given pairwise constraints:

$$\mathcal{L} = \sum_{i,j=1}^{n} \sum_{a,b=1}^{n} S^+_{i,j} S^-_{a,b} \exp(K_{a,b} - K_{i,j}) \qquad (1)$$

In the above, each term within the summation compares $K_{a,b}$, i.e., the similarity between two points from a cannot-link pair, to $K_{i,j}$, i.e., the similarity between two data points from a must-link pair. By minimizing the objective function in (1), we will ensure that all the data points in the must-link pairs are more similar to each other than the data points in the cannot-link pairs.

Input
- X: $d \times n$ matrix for the input data
- $\mathcal{A}$: the given clustering algorithm
- s: the number of principal eigenvectors used for projection
- S^+: matrix for must-link pairs
- S^-: matrix for cannot-link pairs

Output: cluster memberships

Algorithm
- Initialize $K_{i,j} = 0$ for any $i, j = 1, 2, \ldots, n$.
- For $t = 1, 2, \ldots, T$
  - Compute $p_{i,j}$ and $q_{i,j}$ using (5) and (6).
  - Compute matrix T using (10).
  - Compute the top s eigenvectors and eigenvalues $\{(\lambda_i, \mathbf{v}_i)\}_{i=1}^{s}$ of T.
  - Construct the projection matrix P using (11), and generate the new data representation X' by projecting the input data X onto P.
  - Run the clustering algorithm $\mathcal{A}$ using the new data representation X'. Compute the matrix Δ with $\Delta_{i,j} = 1$ when $\mathbf{x}_i$ and $\mathbf{x}_j$ are grouped into the same cluster by $\mathcal{A}$, and zero otherwise.
  - Compute α using (13).
  - Update the kernel similarity matrix K as $K + \alpha\Delta \rightarrow K$
- Run the clustering algorithm $\mathcal{A}$ with the kernel matrix K (if $\mathcal{A}$ does not take a kernel similarity matrix as input, a data representation can be generated by the first $s+1$ eigenvectors of the matrix K).

Figure 2: Boosting algorithm

The objective function in (1) can also be written as:

$$\mathcal{L} = \left(\sum_{i,j=1}^{n} S^+_{i,j} \exp(-K_{i,j}) \right) \left(\sum_{a,b=1}^{n} S^-_{a,b} \exp(K_{a,b}) \right) \qquad (2)$$

The above objective function is a product of two terms: the first term, i.e., $\sum_{i,j=1}^{n} S^+_{i,j} \exp(-K_{i,j})$, measures the inconsistency between the kernel similarity matrix K and the must-link constraints; the second term, i.e., $\sum_{a,b=1}^{n} S^-_{a,b} \exp(K_{a,b})$, measures the inconsistency between the kernel similarity matrix K and the cannot-link constraints. Thus, by minimizing the product of the two terms, we enforce the kernel matrix K to be consistent with the given pairwise constraints.

3.2 The BoostCluster Framework

The overall idea is to improve the clustering results iteratively. In each iteration, we will first identify a new data representation by minimizing the discrepancy between the kernel matrix K and the pairwise constraints. The new data representation will then be used by the clustering algorithm $\mathcal{A}$ to obtain the new clustering results, and the new results will be used in return to update the kernel matrix K. It is important to note that the data representation is generated based on the clustering results. This is where the proposed approach differs from the previous studies, i.e., the proposed algorithm is capable of taking into account the performance

of the clustering algorithm to be boosted while the others do not.

To boost the performance of a clustering algorithm $\mathcal{A}$ given a set of pairwise constraints, we follow the idea of boosting algorithms by iteratively improving the kernel similarity matrix K. Let K denote the current kernel similarity matrix. Let $\Delta \in \{0,1\}^{n\times n}$ denote the incremental kernel similarity matrix that is inferred from the clustering results generated by the algorithm $\mathcal{A}$. In particular, $\Delta_{i,j} = 1$ when both $\mathbf{x}_i$ and $\mathbf{x}_j$ are assigned to the same cluster and $\Delta_{i,j} = 0$ otherwise. The new kernel matrix K' is a linear combination of K and Δ, i.e.,

$$K' \quad = \quad K + \alpha\Delta \tag{3}$$

where $\alpha \geq 0$ is the combination weight. Then, the objective function $\mathcal{L}$ for the new kernel K', denoted by $\mathcal{L}(K')$, is written as:

$$\begin{aligned}\mathcal{L}(K') &= \sum_{i,j=1}^{n}\sum_{a,b=1}^{n} S^{+}_{i,j}S^{-}_{a,b}\exp(K'_{a,b} - K'_{i,j}) \\ &= \sum_{i,j=1}^{n}\sum_{a,b=1}^{n} p_{i,j}q_{a,b}\exp(-\alpha(\Delta_{i,j} - \Delta_{a,b})) \end{aligned} \tag{4}$$

where

$$p_{i,j} \quad = \quad S^{+}_{i,j}\exp(-K_{i,j}) \tag{5}$$

$$q_{a,b} \quad = \quad S^{-}_{a,b}\exp(K_{a,b}) \tag{6}$$

In the above, $p_{i,j}$ measures the inconsistency between the kernel matrix K and the must-link pair $(\mathbf{x}_i, \mathbf{x}_j)$, and $q_{a,b}$ measures the inconsistency between K and the cannot-link pair $(\mathbf{x}_a, \mathbf{x}_b)$.

Using Jensen's inequality, an upper bound for the term $\exp(-\alpha(\Delta_{i,j} - \Delta_{a,b}))$ can be constructed as follows

$$\begin{aligned}&\exp(-\alpha(\Delta_{i,j} - \Delta_{a,b})) \\ &= \exp\left(-3\alpha\frac{\Delta_{i,j} - \Delta_{a,b} + 1}{3} + 3\alpha\frac{1}{3} + 0\times\frac{\Delta_{a,b} - \Delta_{i,j} + 1}{3}\right) \\ &\leq \frac{\Delta_{i,j} - \Delta_{a,b} + 1}{3}\exp(-3\alpha) + \frac{1}{3}\exp(3\alpha) + \frac{\Delta_{a,b} - \Delta_{i,j} + 1}{3}\end{aligned} \tag{7}$$

In the first step of the above derivation, we rewrite $\alpha(\Delta_{i,j} - \Delta_{a,b})$ as a summation over the probability distribution of $((\Delta_{a,b}-\Delta_{i,j}+1)/3, (\Delta_{i,j}-\Delta_{a,b}+1)/3, 1/3)$. Using the upper bound in (7), we can now bound the objective function in (4) as follows

$$\begin{aligned}\mathcal{L}(K') &= \sum_{i,j=1}^{n}\sum_{a,b=1}^{n} p_{i,j}q_{a,b}\exp(-\alpha(\Delta_{i,j} - \Delta_{a,b})) \\ &\leq \frac{\exp(3\alpha) - 1}{3}\sum_{i,j=1}^{n}\Delta_{i,j}\left(p_{i,j}\sum_{a,b=1}^{n} q_{a,b} - q_{i,j}\sum_{a,b=1}^{n} p_{a,b}\right) \\ &\quad + \frac{\exp(3\alpha) + \exp(-3\alpha) + 1}{3}\sum_{i,j=1}^{n}\sum_{a,b}^{n} p_{i,j}q_{a,b}\end{aligned} \tag{8}$$

We can simplify the above expression by defining a matrix T as follows

$$T_{i,j} \quad = \quad \frac{p_{i,j}}{\sum_{a,b=1}^{n} p_{a,b}} - \frac{q_{i,j}}{\sum_{a,b=1}^{n} q_{a,b}}$$

The elements in matrix T measure the inconsistency between kernel matrix K and the pairwise constraints. Only the pairwise constraints that are not well satisfied will resulted in large $|T_{i,j}|$: a large positive value for $T_{i,j}$ indicates that K is inconsistent with the must-link pair $(\mathbf{x}_i, \mathbf{x}_j)$, while a large negative value for $T_{a,b}$ indicates that K is inconsistent with the cannot-link pair $(\mathbf{x}_a, \mathbf{x}_b)$. Here, the matrix T plays a similar role as the example weights $\mathbf{w}$ of the AdaBoost algorithm, in which $\mathbf{w}$ is used to identify the examples that are difficult to classify correctly.

Using the notation of matrix T, the upper bound for $\mathcal{L}$ in (8) becomes

$$\mathcal{L}(K') \leq \mathcal{L}(K) \times \begin{pmatrix} [\exp(3\alpha) + \exp(-3\alpha) + 1]/3 \\ -[1 - \exp(-3\alpha)]\mathrm{tr}(T\Delta^{\top})/3 \end{pmatrix} \tag{9}$$

where

$$\mathcal{L}(K) = \sum_{i,j=1}^{n}\sum_{a,b=1}^{n} p_{i,j}q_{a,b}\ , \quad \mathrm{tr}(T\Delta^{\top}) = \sum_{i,j=1}^{n} T_{i,j}\Delta_{i,j}$$

Note that when $\alpha = 0$, the right side of (9) becomes $\mathcal{L}(K)$, i.e., the objective function of the previous iteration. Thus, by minimizing the upper bound in (9) with respect to α, we are guaranteed to have $\mathcal{L}(K') \leq \mathcal{L}(K)$, thus reducing the objective function at successive iterations.

As suggested by the inequality in (9), to effectively reduce the objective function $\mathcal{L}$, we need to maximize the term $\mathrm{tr}(T\Delta^{\top})$. To obtain the new data representation, we assume that the incremental kernel matrix Δ can be approximated by a linear projection of the input data X, i.e.,

$$\Delta \quad \approx \quad (P^{\top}X)^{\top}(P^{\top}X) = X^{\top}PP^{\top}X$$

where $P = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_s)$ is the projection matrix ($s \leq d$) with each $\mathbf{p}_i \in \mathbb{R}^d$ specifying a different projection direction. Using the above expression, $\mathrm{tr}(T\Delta^{\top})$ can be written as

$$\mathrm{tr}(T\Delta^{\top}) \quad \approx \quad \mathrm{tr}(P^{\top}XTX^{\top}P) \tag{10}$$

By further assuming orthogonality between any two projection vectors, i.e., $\mathbf{p}_i^{\top}\mathbf{p}_j = \delta(i,j)$, we have the optimal solution for $\mathbf{p}_i$ that maximizes the expression in (10) as the ith maximum eigenvector of matrix $XTX^{\top}$. Let $\{(\lambda_i, \mathbf{v}_i)\}_{i=1}^{s}$ denote the top s principal eigenvalues and eigenvectors of matrix $XTX^{\top}$. Then, the optimal projection matrix P is constructed as

$$P \quad = \quad (\sqrt{\lambda_1}\mathbf{v}_1, \sqrt{\lambda_2}\mathbf{v}_2, \ldots, \sqrt{\lambda_s}\mathbf{v}_s) \tag{11}$$

Intuitively, since the matrix T encodes the discrepancy between the current kernel K and the constraints, the projection matrix P, which is generated from the input patterns X and the discrepancy-encoded matrix T, will result in a subspace that best preserves the information from the constraints yet to be satisfied.

Using the projection computed in (11), we generate a new data representation as $X' = P^{\top}X$. This new representation X' will be input to the clustering algorithm $\mathcal{A}$ to generate new cluster memberships. The resulting cluster memberships are then used to compute the incremental kernel matrix Δ.

In addition to the projection matrix P, another important question is how to compute the optimal α. We can estimate the optimal α by minimizing the upper bound in (9), which leads to $\alpha = \log[1 + \mathrm{tr}(T\Delta^{\top})]/6$. However, we can further improve the estimation of α by minimizing the original ob-

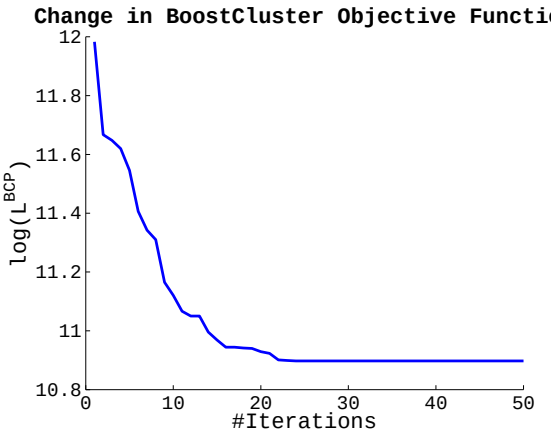


Figure 3: An example of objective function vs. the number of iterations.

jective function in (2), which is

$$\mathcal{L}(K') = \left(\sum_{i,j=1}^{n} S^{+}_{i,j}\exp(-K'_{i,j})\right)\left(\sum_{i,j=1}^{n} S^{-}_{i,j}\exp(K'_{i,j})\right)$$
$$= \left(\sum_{i,j=1}^{n} p_{i,j}\delta(\Delta_{i,j},0) + \sum_{i,j=1}^{n} p_{i,j}\delta(\Delta_{i,j},1)\exp(-\alpha)\right)\times$$
$$\left(\sum_{i,j=1}^{n} q_{i,j}\delta(\Delta_{i,j},0) + \sum_{i,j=1}^{n} q_{i,j}\delta(\Delta_{i,j},1)\exp(\alpha)\right) \quad (12)$$

It is not difficult to show that the optimal α that maximizes the above expression is:

$$\alpha = \frac{1}{2}\log\left(\frac{\sum_{i,j=1}^{n} p_{i,j}\delta(\Delta_{i,j},1)}{\sum_{i,j=1}^{n} p_{i,j}\delta(\Delta_{i,j},0)} \times \frac{\sum_{i,j=1}^{n} q_{i,j}\delta(\Delta_{i,j},0)}{\sum_{i,j=1}^{n} q_{i,j}\delta(\Delta_{i,j},1)}\right) \quad (13)$$

Figure 2 summarizes the proposed boosting algorithm.

3.3 Convergence

Similar to most boosting algorithms, we can show that the objective function (1) is reduced exponentially with successive iterations of the proposed boosting algorithm. This conclusion can be summarized into the following theorem.

THEOREM 1. *Let $\Delta^1, \Delta^2, \ldots, \Delta^T$ be the incremental kernel matrices computed from the clustering results by running the boosting algorithm (in Figure 2). Then, the objective function after T iterations, i.e., $\mathcal{L}_T$, is bounded as follows:*

$$\mathcal{L}_T \leq \left(\sum_{i,j=1}^{n} S^{+}_{i,j}\right)\left(\sum_{i,j=1}^{n} S^{-}_{i,j}\right)\prod_{t=1}^{T}(1-\gamma_t), \quad (14)$$

where

$$\gamma_t = \frac{(\sqrt{A_t D_t} - \sqrt{B_t C_t})^2}{(A_t + B_t)(C_t + D_t)}$$

A_t, B_t, C_t, and D_t are non-negative constants, and are computed as

$$A_t = \sum_{i,j=1}^{n} p^t_{i,j}\delta(\Delta^t_{i,j},0),\ B_t = \sum_{i,j=1}^{n} p^t_{i,j}\delta(\Delta^t_{i,j},1)$$
$$C_t = \sum_{i,j=1}^{n} q^t_{i,j}\delta(\Delta^t_{i,j},0),\ D_t = \sum_{i,j=1}^{n} q^t_{i,j}\delta(\Delta^t_{i,j},1)$$

where $p^t_{i,j}$ and $q^t_{i,j}$ are computed according to (5) and (6) using the kernel matrix K at the t-th iteration.

The above theorem can be proved directly by using the expression in (12) and the expression for α in (13). Due to space constraints, we cannot provide details. Figure 3 shows how the logarithm of the objective function used by the proposed algorithm changes with respect to the number of iterations; the objective function converges very fast, and becomes flat after 22 iterations. In our experiments, the boosting algorithm usually converges within 25 iterations.

3.4 Computational Issues

In the proposed boosting algorithm, a key step towards finding a good projection matrix P is eigen-decomposition of $XTX^\top$, as shown in (10) and (11). To efficiently compute $XTX^\top$, we first note that $XTX^\top$ can also be written as:

$$XTX^\top = \sum_{i,j=1}^{n} T_{i,j}\mathbf{x}_i\mathbf{x}_j^\top \quad (15)$$

Since $T_{i,j}$ is nonzero only when the example pair $(\mathbf{x}_i, \mathbf{x}_j)$ corresponds to a given constraint, the above calculation only involves a very small portion of all possible example pairs. Hence, the computational cost for $XTX^\top$ is only proportional to the number of pairwise constraints. Therefore, $XTX^\top$ can be calculated efficiently as long as the number of labeled pairs is relatively small.

In addition to $XTX^\top$, another major computational cost arises from calculating the principal eigenvectors and eigenvalues of $XTX^\top$, particularly when the dimensionality of the feature space is high. For instance, for text data clustering, each document is typically represented by a vector of over 100,000 features, and the size of matrix $XTX^\top$ is over $100,000 \times 100,000$. A straightforward approach is to reduce the dimensionality before running the proposed algorithm. However, most dimensionality reduction algorithms that are capable of handling high dimensional space are unsupervised, and therefore are unable to exploit the pairwise constraints. Here, we propose an algorithm that is able to efficiently compute the eigenvectors of $XTX^\top$ when the input dimensionality is high. We first realize that each eigenvector of $XTX^\top$ has to lie in the space that is spanned by the examples used by the constraints. More specifically, we denote by $\tilde{X} = (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_m)$ the subset of m examples that are involved in the constraints. Then, the eigenvector $\mathbf{v}_i$ can be written as a linear combination of $\{\tilde{\mathbf{x}}_i\}_{i=1}^m$, i.e.,

$$\mathbf{v}_i = \sum_{k=1}^{m} w_{i,k}\tilde{\mathbf{x}}_k = \tilde{X}\mathbf{w}_i \quad (16)$$

More generally, we have

$$V = (\mathbf{v}_1, \mathbf{v}_2 \ldots, \mathbf{v}_s) = \tilde{X}(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_s) = \tilde{X}W.$$

The proof of this result can be found in Appendix A. We furthermore denote by $\tilde{T}$ the pairwise constraints, where $\tilde{T}_{i,j}$ denotes the pairwise constraint between $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$. Then, $\mathbf{w}_i, i = 1, 2, \ldots, s$ correspond to the first s principal eigenvectors of the following generalized eigenvector problem

$$\tilde{X}^\top\tilde{X}\tilde{T}\tilde{X}^\top\tilde{X}\mathbf{w}_i = \lambda_i\tilde{X}^\top\tilde{X}\mathbf{w}_i \quad (17)$$

Note that since $\tilde{X}^\top\tilde{X}\tilde{T}\tilde{X}^\top\tilde{X}$ and $\tilde{X}^\top\tilde{X}$ are $m \times m$ matrices, the cost of computing the eigenvectors is independent of the dimensionality of the input space. The proof of the above result can be found in Appendix B.

Name	#Attributes	#Clusters	#Examples
wdbc	30	2	569
scale	4	3	625
vowel	10	11	990
segmentation	19	7	2310
handwrittendigit	256	10	4000
pendigit	16	4	4396

Table 1: Datasets used in the experiments.

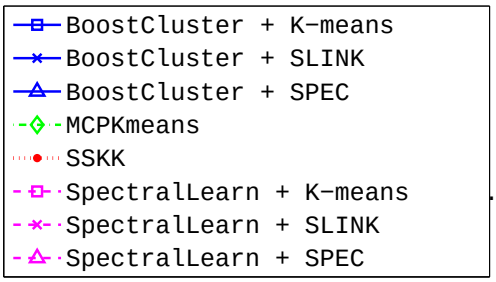


Figure 4: Legends for all algorithms in our comparative study. These legends will be used in all the figures in this paper.

4. EXPERIMENTS

We now present an empirical evaluation of our proposed boosting framework. In particular, we aim to address the following three questions in our study:

1. *As a general boosting framework, is the proposed method able to improve the performance for any given clustering algorithm*?
2. *How effective is the proposed boosting framework compared to other semi-supervised clustering algorithms*?
3. *How robust is the proposed boosting framework in improving the clustering performance by using the pairwise constraints*?

4.1 Experiment Setup

To validate the claim that the proposed boosting algorithm is capable of improving *any* clustering algorithm by exploiting the pairwise constraints, three popular clustering algorithms are used in our study. They are:

1. K-means algorithm [1]. It represents the family of clustering algorithms that try to find compact and well-separated clusters. We adopted the implementation from the Weka software[1].
2. Partitional SingleLink algorithm ("SLINK" for short) [16]. It represents the family of the hierarchical clustering algorithms. We adopted the implementation from the CLUTO software[2]
3. k-way spectral clustering ("SPEC" for short). It represents the family of spectral methods for data clustering. In particular, we follow the algorithm in [24] for the implementation of spectral clustering.

Six datasets drawn from the UCI machine learning repository [9] are used in our study. Table 1 summarizes the information about these datasets[3]. As indicated in Table 1, these datasets vary significantly in their sizes, number of clusters, and number of attributes.

To evaluate the clustering performance, two measurements are used in our experiments. The first measurement is normalized mutual information (**NMI** for short) [4], which is defined as

$$NMI = \frac{2MI(X, X_0)}{H(X) + H(X_0)}$$

where X_0 and X denote the random variables of cluster memberships from the ground truth and the output of clustering algorithm, respectively. $MI(x, y)$ represents the mutual information between random variables x and y, and $H(x)$ represents the Shannon entropy of random variable x. The second measurement is Pairwise F-measure (**PWF1** for short), which is the harmonic mean of pairwise *precision* and *recall* that are defined as follows

$$precision = \frac{\#\text{pairs correctly placed in the same cluster}}{\text{Total } \#\text{pairs placed in the same cluster}}$$

$$recall = \frac{\#\text{pairs correctly placed in the same cluster}}{\text{Total } \#\text{pairs actually in the same cluster}}$$

$$PWF1 = \frac{2 \times precision \times recall}{precision + recall}$$

The PWF1 measurement defined above is closely related to the metric defined in [29] that measures the percentage of data pairs correctly clustered together. The key problem with the metric defined in [29] is that it counts two types of data pairs, i.e., pairs of data points assigned to the same cluster and pairs of data points assigned to different clusters, with equal importance. This is problematic because most of the data pairs in practice will consist of data points from different clusters when the number of clusters is large. A similar issue arises in multi-class learning, and that is why F-measure is widely used for evaluating multi-class learning [31].

To verify the efficacy of the proposed boosting framework in exploiting the pairwise constraints for data clustering, three baseline approaches are used:

1. *Metric Pairwise Constraints K-means* (**MPCKmeans** for short) algorithm [2,4], which is a probabilistic framework based on Hidden Markov Random Fields.
2. *Semi-supervised Kernel K-means* (**SSKK** for short) algorithm [19], which exploits the pairwise constraints by a kernel approach and finds clusters with nonlinear boundaries in the input data space.
3. *Spectral Learning* (**SpectralLearn** for short) algorithm [17], which applies spectral methods to learn a data representation from the pairwise constraints. The generated data representation can therefore be used by *any* clustering algorithm to identify the appropriate data clusters. The key difference between spectral learning and our algorithm is that our algorithm generates algorithm specific data representations by taking into account the performance of clustering algorithms.

[1] `http://www.cs.waikato.ac.nz/ml/weka/`
[2] `http://glaros.dtc.umn.edu/gkhome/views/cluto`
[3] Note that for the "pendigit" dataset, examples in only four classes of letter "3", "6", "8" and "9" are selected from a total of 10 classes because these four letters are in general most difficult to distinguish.

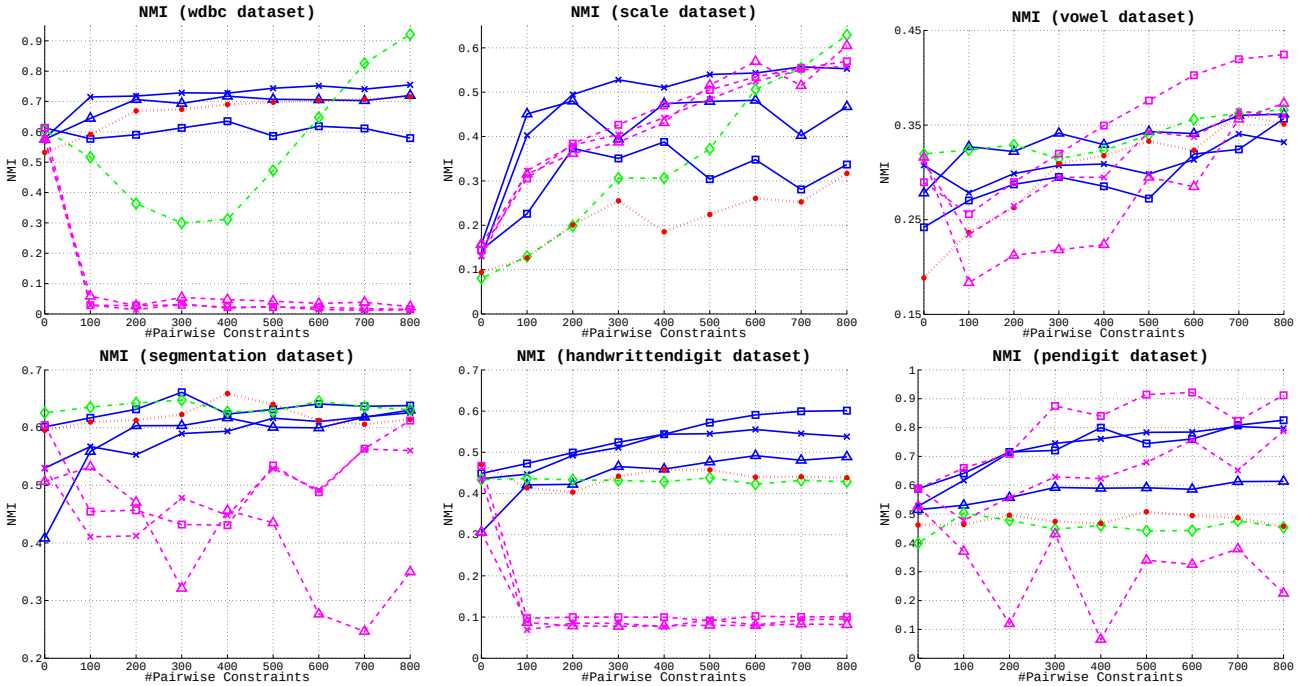


Figure 5: Comparison of clustering performance evaluated by NMI.

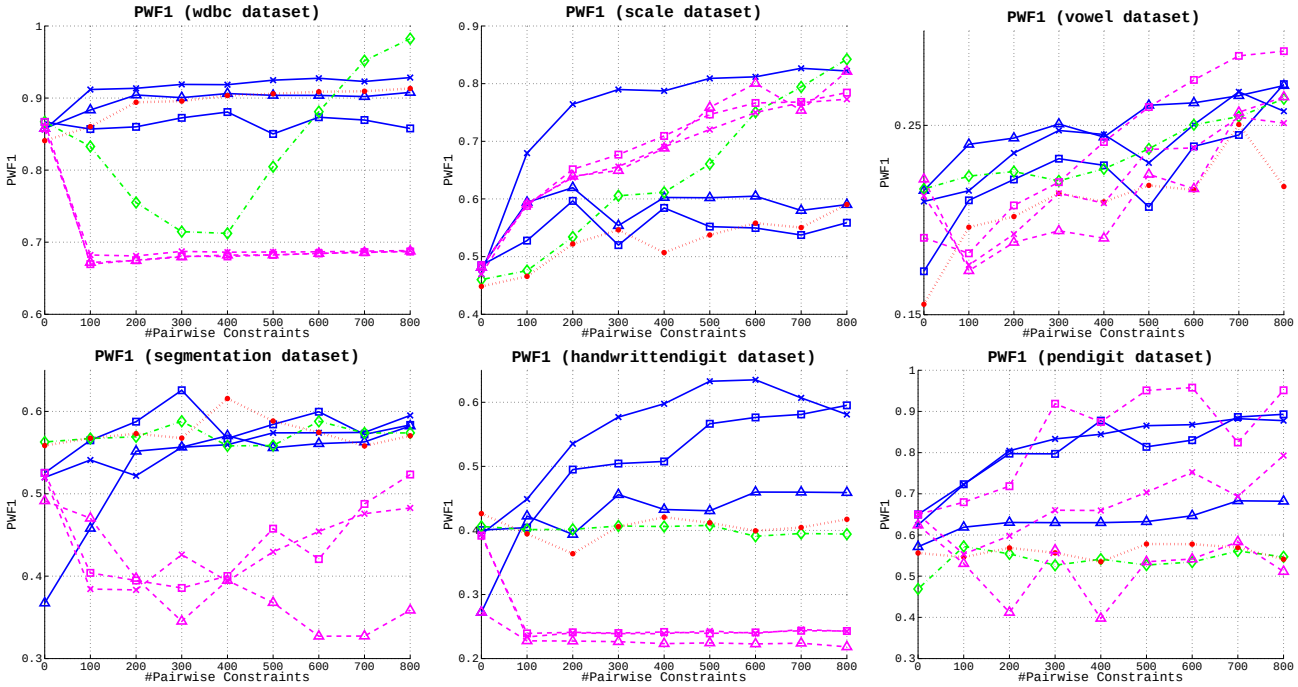


Figure 6: Comparison of clustering performance evaluated by PWF1.

Previous studies [2, 4, 17, 19] showed that the above three algorithms deliver the state-of-the-art performance in comparison to other semi-supervised clustering algorithms such as the constrained K-means.

In summary, we will compare the following semi-supervised clustering algorithms in the experiments: the three clustering algorithms (K-means, SLINK, and SPEC) being boosted by the proposed BoostCluster framework; the same three clustering algorithms with input from the Spectral Learning algorithm; the MPCKmeans algorithm; and the SSKK algorithm. For easy identification in figures, we listed the legends for the above algorithms to be compared, in Figure 4. These legends apply to all following performance comparison figures (we will omit showing legends in those figures due to space constraints).

Finally, in all the experiments, we vary the number of pairwise constraints from 0 to 800. Since a random sampling of data pairs tends to find many more cannot-link pairs than the must-link pairs, in this study, we enforce an equal number of constraints for both must-link pairs and cannot-link pairs. The numbers of eigenvectors (i.e., the parameter s in the boosting algorithm shown in Figure 2) are determined empirically as follows: 3 for the "scale" dataset, 10 for the "handwrittendigit" dataset and 5 for the remaining 4 datasets. All the experiments in this study are repeated five times, and the evaluation results averaged over the five trials are reported.

4.2 Generality of the Boosting Framework

Figures 5 and 6 show the clustering performance, evaluated by NMI and PWF1 respectively, of the BoostCluster framework using the three clustering algorithms (i.e., K-means, partitional SingleLink, and spectral clustering), the same three clustering algorithms with input as the new data representation from the Spectral Learning algorithm, the MPCKmeans algorithm, and the SSKK algorithm.

1. We observe that for most datasets, the BoostCluster framework is able to improve the clustering performance for all the three clustering algorithms regardless of which evaluation metric is used. This suggests that the proposed framework is effective in exploiting the pairwise constraints to improve clustering performance. MPCKmeans algorithm and SSKK algorithm are also effective in general, however, their clustering performance improvements are less significant, especially for larger datasets (such as "handwrittendigit" and "pendigit").
2. Although SpectralLearn algorithm can also be combined with *any* clustering algorithm, in our experiments, it does not always improve the clustering algorithm performance. For example, for "wdbc" and "handwrittendigit", increasing the number of pairwise constraints deteriorates clustering performance by combining SpectralLearn with any of the three clustering algorithms. Moreover, the effect of SpectralLearning depends on the clustering algorithm. For example, for the "pendigit" dataset, SpectralLearn improves the clustering performance of K-means and SLINK, but degrades SPEC in general. In comparison, the clustering performance improvement brought by the proposed BoostCluster is substantially more stable and consistent, across different datasets and different clustering algorithms. This can be attributed to the fact that BoostCluster is adaptive to both clustering algorithms and datasets: in each iteration, it takes the feedback from the result of applying the given clustering algorithm to the particular dataset, and decides how to adjust the kernel matrix. However, SpectralLearn generates new data representations independent from the clustering algorithm that is used.
3. The performance of the three clustering algorithms (K-means, SLINK, and SPEC) varies significantly across the six different datasets. For instance, for the "vowel" dataset, "BoostCluster+K-means" algorithm performs considerably worse than "BoostCluster+SPEC" algorithm. However, the performance of "BoostCluster+K-means" algorithm for the "pendigit" dataset, is significantly better than that of "BoostCluster+SPEC" algorithm. This result also indicates that every clustering algorithm has its own strength, and therefore it is important to develop a general framework that is able to boost the performance of any clustering algorithm by the given pairwise constraints.
4. The results based on the two different evaluation metrics, namely NMI and PWF1, are inconsistent on some occasions. For instance, for the "handwrittingdigit" dataset, according to NMI, the clustering performance of "BoostCluster+K-means" and "BoostCluster+SLINK" appears to be similar. However, according to PWF1, "BoostCluster+SLINK" performs noticeably better than "BoostCluster+K-means". The implication of this finding is the importance of evaluating clustering performance by more than one evaluation metric, since conclusions based on the results of a single evaluation metric could be biased.

4.3 Robustness of Exploiting Pairwise Constraints

Although the curves in Figures 5 and 6 all display different degrees of "bumpiness", generally speaking, BoostCluster framework, SSKK and MPCKmeans deliver a more robust performance than SpectralLearn algorithm. On the other hand, although for most datasets, the performance curves of SSKK appear to be the most smooth among all the competitors, the resultant improvement is almost always the least noticeable among all the semi-supervised clustering algorithms.

To further evaluate the robustness of all the algorithms, we conduct experiments with noisy pairwise constraints. We randomly select 20% of the pairwise constraints and flip their labels (i.e., a must-link pair becomes a cannot-link pair and vice versa). This setting reflects the scenario when the side information includes incorrect pairwise constraints. It could happen when for instance, the pairwise constraints are derived from the implicit user feedback (e.g., user ratings or click-through data). Thus, it is important to develop semi-supervised clustering algorithms that are resilient to the noisy side information.

Figure 7 shows the performance of all the algorithms, on three selected datasets (i.e., "scale", "vowel", and "pendigit") when 20% of the pairwise constraints are noisy [4]. First, by comparing Figures 7 with Figures 5 and 6, it is not sur-

[4]Due to space constraints, we are unable to show the results for all the six datasets.

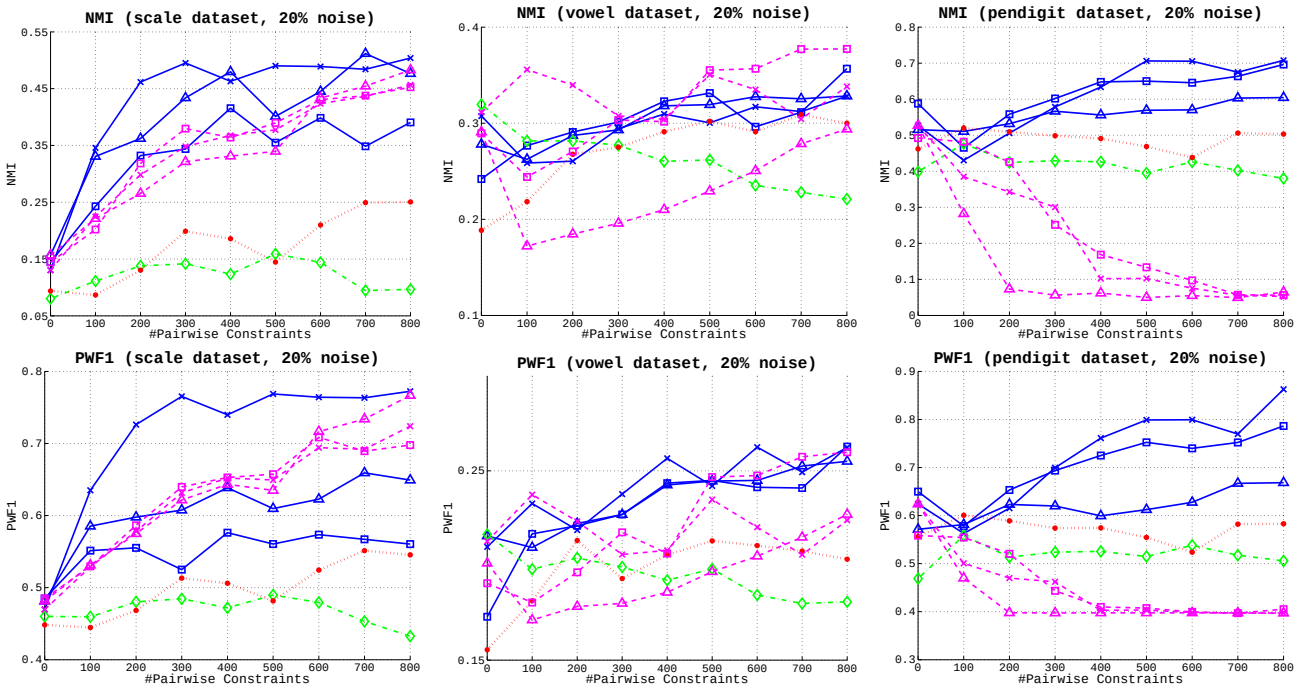


Figure 7: Comparison of clustering performance with 20% noise in the pairwise constraints. Graphs in the top row show the NMI measurements, and graphs in the lower row show the PWF1 measurements.

prising to observe a degradation in clustering performance when 20% of the pairwise constraints are noisy. Second, we observe a general trend that a larger number of noisy constraints tend to result in an inferior clustering performance by MPCKmeans. This is in contrast to the results of MPCKmeans shown in Figures 5 and 6 where increasing the number of pairwise constraints usually improves the performance of clustering. This result implies that the MPCKmeans algorithm is unable to effectively exploit the pairwise constraints for data clustering when they are noisy. Similarly, while "SpectralLearn+K-means" and "SpectralLearn+SLINK" are able to noticeably improve the clustering performance with increasing number of noise-free pairwise constraints, with 20% noise in the constraints, their performance degrades with the increasing number of constraints. In comparison, as shown in Figure 7, BoostCluster framework is generally able to improve the performance of all the three clustering algorithms with increasing number of noisy pairwise constraints, and SSKK algorithm is also able to improve clustering performance despite the noise in pairwise constraints. This indicates that the proposed BoostCluster framework and the SSKK algorithm are more resilient to the noise in the side information.

5. CONCLUSIONS

In this paper, we have studied the problem of improving data clustering by using side information in the form of pairwise constraints. A general boosting framework has been proposed to improve the accuracy of *any* given clustering algorithm with a given set of pairwise constraints. Such performance improvement is achieved by iteratively finding new data representations that are consistent with both the clustering results from previous iterations and the pairwise constraints. Empirical study shows that our proposed boosting framework is able to improve the clustering performance of several popular clustering algorithms by using the pairwise constraints.

6. ACKNOWLEDGMENTS

This research work is supported by NSF grant IIS-0643494, NIH grants 1R01GM079688-01 and 1R21NS054148-01A1, and ONR grant N00D140710225. We would also like to thank anonymous reviewers for their valuable suggestions.

7. REFERENCES

[1] M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, Inc., New York, NY, 1973.

[2] S. Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, The University of Texas at Austin, 2005.

[3] S. Basu, A. Banerjee, and R. J. Mooney. Active semi-supervision for pairwise constrained clustering. In *ICDM '04*, 2004.

[4] S. Basu, M. Bilenko, and R. J. Mooney. A probabilistic framework for semi-supervised clustering. In *SIGKDD '04*, 2004.

[5] R. Bekkerman and M. Sahami. Semi-supervised clustering using combinatorial MRFs. In *ICML-06 Workshop on Learning in Structured Output Spaces*, 2006.

[6] K. Bennett, P. Bradley, and A. Demiriz. Constrained k-means clustering. Technical Report 2000-65, Microsoft Research, May 2000.

[7] I. Davidson and S. Ravi. Clustering under constraints: Feasibility results and the k-means algorithm. In *SIAM Data Mining Conference*, 2005.

[8] I. Davidson and S. Ravi. Hierarchical clustering with constraints: Theory and practice. In *the 9th European Principles and Practice of KDD (PKDD)*, 2005.

[9] C. B. D.J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases, 1998.

[10] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[11] J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *NIPS'04*, 2004.

[12] M. Halkidi, D. Gunopulos, N. Kumar, M. Vazirgiannis, and C. Domeniconi. A framework for semi-supervised learning based on subjective and objective clustering criteria. In *ICDM '05*, 2005.

[13] T. Hertz, A. Bar-Hillel, and D. Weinshall. Boosting margin based distance functions for clustering. In *ICML '04*, 2004.

[14] T. Hertz, A. B. Hillel, and D. Weinshall. Learning a kernel function for classification with small training samples. In *ICML '06*, 2006.

[15] S. C. H. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma. Learning distance metrics with contextual constraints for image retrieval. In *CVPR '06*, 2006.

[16] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, September 1999.

[17] S. D. Kamvar, D. Klein, and C. D. Manning. Spectral Learning. In *IJCAI '03*, 2003.

[18] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML'02*, 2002.

[19] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: a kernel approach. In *ICML '05*, 2005.

[20] J. T. Kwok and I. W. Tsang. Learning with idealized kernels. In *ICML '03*, 2003.

[21] T. Lange, M. H. C. Law, A. K. Jain, and J. M. Buhmann. Learning with constrained and unlabelled data. In *CVPR '05*, 2005.

[22] M. H. C. Law, A. P. Topchy, and A. K. Jain. Model-based clustering with probabilistic constraints. In *SDM '05*, 2005.

[23] Z. Lu and T. Leen. Semi-supervised learning with penalized probabilistic clustering. In *NIPS '05*, 2005.

[24] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS '01*, 2001.

[25] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *NIPS '03*, 2003.

[26] N. Shental, A. Bar-Hillel, T. Hertz, and D. Weinshall. Computing gaussian mixture models with EM using side-information. In *Proc. of workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, 2003.

[27] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *ICML '01*, 2001.

[28] K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS '06*, 2006.

[29] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *NIPS '03*, 2003.

[30] L. Yang, R. Jin, R. Sukthankar, and Y. Liu. An efficient algorithm for local distance metric learning. In *AAAI '06*, 2006.

[31] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR '99*, 1999.

[32] Z. Zhang, J. Kwok, and D. Yeung. Parametric distance metric learning with label information. In *IJCAI'03*, 2003.

APPENDIX

A. PROOF 1

We show that every non-zero eigenvector $\mathbf{v}_i$ can be written as a linear combination of $\tilde{\mathbf{x}}_i, i = 1, 2, \ldots, m$, i.e., the examples involved in the pairwise constraints. Let $\mathbf{v}$ and $\lambda \neq 0$ be an eigenvector and eigenvalue of matrix $XTX^\top$. We therefore have $XTX^\top\mathbf{v} = \lambda\mathbf{v}$. We further decompose $\mathbf{v}$ into two parts: $\mathbf{v} = \mathbf{v}_\parallel + \mathbf{v}_\perp$, where $\mathbf{v}_\parallel$ represents the projection of $\mathbf{v}$ in the subspace spanned by $\{\tilde{\mathbf{x}}_i\}_{i=1}^s$, and $\mathbf{x}_\perp$ represents the projection perpendicular to $\{\tilde{\mathbf{x}}_i\}_{i=1}^s$. To show that $\mathbf{v}$ can be written as a linear combination of $\{\tilde{\mathbf{x}}_i\}_{i=1}^s$, we need to show $\mathbf{v}_\perp = \mathbf{0}$. To this end, we first utilize the expression in (15) to calculate $(\mathbf{v}_\perp)^\top XTX^\top$, i.e.,

$$(\mathbf{v}_\perp)^\top XTX^\top = \sum_{i,j=1}^{m} \tilde{T}_{i,j}(\mathbf{v}_\perp)^\top \tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_j^\top = \mathbf{0}^\top$$

We then multiply the eigen equation $XTX^\top\mathbf{v} = \lambda\mathbf{v}$ by $(\mathbf{v}_\perp)^\top$, which leads to the following equation

$$(\mathbf{v}_\perp)^\top XTX^\top\mathbf{v} = 0 = \lambda(\mathbf{v}_\perp)^\top\mathbf{v} = \lambda\|\mathbf{v}_\perp\|_2^2$$

Since $\lambda \neq 0$, we have $\mathbf{v}_\perp = \mathbf{0}$ and $\mathbf{v} = \mathbf{v}_\parallel$.

B. PROOF 2

We will show that for the ith eigenvector $\mathbf{v}_i = \tilde{X}\mathbf{w}_i$ of $\tilde{X}\tilde{T}\tilde{X}^\top$, $\mathbf{w}_i$ corresponds to the ith eigenvector of the generalized eigenvector problem in (17). First, we realize that the orthogonality condition $\mathbf{v}_i^\top\mathbf{v}_j = \delta(i,j)$ becomes $\mathbf{w}_i^\top\tilde{X}\tilde{X}^\top\mathbf{w}_j = \delta_{i,j}$. We can write the above condition for all $\mathbf{w}_i, i = 1, 2, \ldots, s$ in the matrix form, i.e., $W^\top\tilde{X}\tilde{X}^\top W = I_s$. Second, the eigenvectors $V = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_s)$ are the optimal solution to the following optimization problem, i.e.,

$$\begin{aligned} \arg\max_{V\in\mathbb{R}^{d\times s}} \quad & \mathrm{tr}(V^\top XTX^\top V) \\ \text{s. t.} \quad & V^\top V = I_s \end{aligned}$$

Replacing V in the above optimization problem with $V = \tilde{X}W$, we have

$$\begin{aligned} \max_{W\in\mathbb{R}^{m\times s}} \quad & \mathrm{tr}(W^\top\tilde{X}^\top\tilde{X}\tilde{T}\tilde{X}^\top\tilde{X}W) \\ \text{s. t.} \quad & W^\top\tilde{X}^\top\tilde{X}W = I_s \end{aligned}$$

It is well known that the optimal solution W to the above problem consists of the first s eigenvectors of the generalized eigenvector problem in (17).