Nonnegative Sparse Coding for Discriminative Semi-supervised Learning

Ran He[1], Wei-Shi Zheng[2] , Bao-Gang Hu[1], Xiang-Wei Kong[3]
[1]NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
{rhe, hubg}@nlpr.ia.ac.cn
[2] School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510275, China
wszheng@ieee.org
[3] Department of Electronic Engineering, Dalian University of Technology, Dalian 116024, China
kongxw@dlut.edu.cn

Abstract

An informative and discriminative graph plays an important role in the graph-based semi-supervised learning methods. This paper introduces a nonnegative sparse algorithm and its approximated algorithm based on the l^0-l^1 equivalence theory to compute the nonnegative sparse weights of a graph. Hence, the sparse probability graph (SPG) is termed for representing the proposed method. The nonnegative sparse weights in the graph naturally serve as clustering indicators, benefiting for semi-supervised learning. More important, our approximation algorithm speeds up the computation of the nonnegative sparse coding, which is still a bottle-neck for any previous attempts of sparse non-negative graph learning. And it is much more efficient than using l^1-norm sparsity technique for learning large scale sparse graph. Finally, for discriminative semi-supervised learning, an adaptive label propagation algorithm is also proposed to iteratively predict the labels of data on the SPG. Promising experimental results show that the nonnegative sparse coding is efficient and effective for discriminative semi-supervised learning.

1. Introduction

Graph-based discriminative semi-supervised learning [21] typically relies on a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ to represent the whole data set, where the $\mathscr{V}$ is the vertex set with respect to the data set and the $\mathscr{E}$ is the edge set associated with a set of non-negative weights. A graph represents the pairwise relationship between data. From a machine learning perspective, three characteristics of an informative graph are often desirable [22]: high discriminating power, sparsity, and adaptive neighborhood.

Though millions of such a kind of relationship can be explored from data, recent studies on sparse signal representation [6, 23, 22] have suggested that performing selection of pairwise relationship is critical for pattern analysis, and therefore the l^1-graph model [24, 9, 5] whose weights are computed via sparse data coding has been widely studied.

This paper presents a novel graph model, named sparse probability graph (SPG), to address the above three basic characteristics of graph models. SPG assumes that each sample can be linearly reconstructed by a sparse probability representation of the training data. Though the non-negativity constraint has not been considered for sparse graph learning, it has been widely considered in many aspects in other machine learning, pattern recognition and computer vision models [14, 20, 10, 11]. Incorporating the non-negativity constraint into sparse graph learning benefits for learning the probabilistic latent clustering relationship between data. Moreover, as we will discuss later, the non-negativity based sparse graph learning is also helpful for sparse data such as document database in text classification.

To compute the nonnegative sparse weights of SPG, we develop efficient nonnegative sparse algorithms based on the l^0-l^1 equivalence theory. Our algorithms result in a speedup for nonnegative sparse coding, allowing us to learn larger sparse codes than l^1 graph. Then an adaptive label propagation algorithm is proposed to iteratively predict the labels of data on the SPG for discriminative semi-supervised learning. Extensive experimental results demonstrate that nonnegative sparse coding is also efficient and effective for graph-based semi-supervised learning, and SPG is more sparse and informative than the l^1 graph.

The remainder of this paper is organized as following: in Section 2, we begin with a brief review of sparse code algorithms and nonnegative sparse code algorithms. Then we propose an l^1 regularized nonnegative sparse coding algorithm and an efficient nonnegative sparse coding algorithm in Section 3. Based on nonnegative sparse coding, we proposed our sparse probability graph algorithm for discriminative semi-supervised learning in Section 4. In Section 5, we compared the SPG with the state-of-the-art graph based algorithms w.r.t. classification rate, sparsity, and computational cost. Finally, we draw the conclusions in Section 6.

2. Sparse Coding Algorithms

Let $X \doteq [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{d \times n}$ be a matrix whose columns are n training samples, and x_{ik} be the k-th entry of x_i. Let $X_{\hat{i}}$ be the matrix obtained from X by removing its i-th column x_i.

2.1. Sparse Coding Algorithms

In machine learning and computer vision, one aims to seek the a suitable sparse solution from the whole training set X, i.e.,

$$\min ||\beta||_0 \quad s.t. \quad y = X\beta \tag{1}$$

where $||.||_0$ denotes the l^0-norm, which counts the number of nonzero entries in a vector, and $y \in \mathbb{R}^{d\times 1}$ is an input sample. However, the problem of finding the sparse solution of Eq.(1) is NP-hard, and difficult to solve.

The theory of compressive sensing [4, 6] reveals that if the solution β is sparse enough, we can solve the following convex relaxed optimization problem to obtain approximate solution

$$\min ||\beta||_1 \quad s.t. \quad y = X\beta \tag{2}$$

where $||.||_1$ denotes the l^1-norm. Here, we denote the algorithm using Eq.(2) to construct a graph by l^1**-graph0**.

To deal with occlusions and corruptions, [23] further proposed a robust linear model as $y = X\beta + e$ where $e \in \mathbb{R}^d$ is a noise item of errors. Assuming that the noise item e has also a sparse representation, the l^1-graph can compute a robust weight vector w_i as following:

$$\min ||\beta||_1 + ||e||_1 \quad s.t. \quad ||y - (X\beta + e)||_2 \le \varepsilon \tag{3}$$

We denote the algorithm using Eq.(3) to construct a graph by l^1**-graph1**.

However, in many applications the noise level ε is unknown beforehand [9]. In such cases the Lasso optimization algorithm [19] can be used to recover the sparse solution from

$$\min ||y - (X\beta + e)||_2^2 + \lambda(||\beta||_1 + ||e||_1) \tag{4}$$

where λ can be viewed as an inverse of the Lagrange multiplier in Eq.(3). Note that the model in Eq.(3) and model in Eq.(4) are different when there is noise [23]. The ε can be interpreted as a pixel noise level, whereas λ cannot [23]. Although (3) and (4) try to solve the same optimization problem, their performance is different for real-world problems [25]. We denote the algorithm using Eq.(4) to construct a graph by l^1**-graph2**.

Although l^1-graph indeed improves the classification rate of discriminative semi-supervised learning against traditional methods in most cases[24, 22, 5], it becomes less informative when data is sparse.

2.2. Nonnegative Sparse Coding Algorithms

In machine learning and computer vision, one also aims to seek a nonnegative sparsest solution from the whole training set X [1], i.e.,

$$\min ||\beta||_0 \quad s.t. \quad y = X\beta \quad and \quad \beta \ge 0 \tag{5}$$

The problem of finding the sparse solution of Eq.(5) is NP-hard [7, 3], and very hard to solve in general. Fortunately, we could replace the l^0-norm by an l^1-norm [7, 3] if the solution β is sparse enough. Then we can solve the following linear programming problem to obtain an approximate solution

$$\min ||\beta||_1 \quad s.t. \quad y = X\beta \quad and \quad \beta \ge 0 \tag{6}$$

The orthogonal matching pursuit algorithm [3], second-order cone programming [14] and nonnegative least squares [20][10] were proposed to solve Eq.(6). [11] combines nonnegative sparse coding and maximum correntropy criterion [16] to deal with occlusion and corruption for robust face recognition. Although extensive experimental observations [20][10][11] show that without harnessing the l^1-norm technique the non-negative least squares technique can also learn a sparse representation for image-based object recognition, a theoretical investigation still needs to be further done to support the sparse idea and discuss its relationship with the l^1 minimization technique [20][10]. Moreover, finding sparse codings and nonnegative sparse codings remain difficult computational problems, which is a widely open topic [22][8].

3. Nonnegative Sparse Coding Algorithms

In this section, we firstly propose an l^1 regularized nonnegative sparse coding algorithm based on the l^0-l^1 equivalence theory [7, 3]. Then an efficient nonnegative sparse coding algorithm is proposed based on the analysis of the effectiveness of the l^1 regularized item.

3.1. l^1 Regularized Nonnegative Sparse Coding Algorithm

The nonnegative sparse coding algorithm aims to find the sparsest solution of an underdetermined and nonnegative linear system. Based on the l^0-l^1 equivalence theory and Lagrange multiplier method, we can rewrite Eq.(6) as

$$\min_{\beta} ||y - X\beta||_2^2 + \lambda||\beta||_1 \quad s.t. \quad \beta \ge 0 \tag{7}$$

The optimal problem in Eq.(7) can be re-formulated as the following quadratic program:

$$\min_{\beta} (\tfrac{\lambda}{2} - X^T y)^T \beta + \tfrac{1}{2}\beta^T X^T X \beta \quad s.t. \quad \beta \ge 0 \tag{8}$$

[1] Different from learning the nonnegative sparse coding of unknown components [12][13], this work assumes that the components X are known and only focuses on nonnegative sparse representation.

Since X^TX is a positive semidefinite matrix, this quadratic program in (8) is convex. Based on the Karush-Kuhn-Tucker optimal conditions, the following monotone linear complementary problem (LCP) is derived [18] [2]:

$$\alpha = X^TX\beta - X^Ty + \tfrac{\lambda}{2}, \quad \alpha \geq 0,\ \beta \geq 0,\ \beta^T\alpha = 0 \quad (9)$$

Algorithm 1 l^1 Regularized Nonnegative Sparse Coding Algorithm

```
1: Input: Data matrix $X$, test sample $y$, $F = \phi$, $G = \{1,\ldots,n\}$, $\beta = \mathbf{0}$, and $\alpha = -X^Ty$.
2: Output: sparse code $\beta$.
3: Normalize the columns of $X$ and $y$ to have unit $l^2$-norm.
4: Compute $r = \arg\min\{\alpha_i : i \in G\}$. If $\alpha_r < 0$, set $F = F \cup r$, $G = G - r$.
   Otherwise stop: $\beta^* = \beta$ is the optimal solution.
5: Compute $\bar{\beta}_F$ by solving (10). If $\bar{\beta}_F \geq 0$, set $\beta^t = (\bar{\beta}_F, 0)$ and go to step 4.
   Otherwise let $r$ be such that:
       $\theta = \frac{-\beta_r}{\bar{\beta}_r - \beta_r} = \min\{\frac{-\beta_i}{\bar{\beta}_i - \beta_i} : i \in F \ and\ \bar{\beta}_i < 0\}$
   and set $\beta^t = ((1-\theta)\beta_F + \theta\bar{\beta}_F,\ 0)$, $F = F - r$,
   $G = G \cup r$. Return to step 2.
6: Compute $\alpha$ according to (11) and return to step 1.
```

If the matrix X has full column rank ($rank(X) = n$), the convex program in (8) and the LCP in (9) have unique solutions for each vector y .

Let F and G be two subsets of $\{1,\ldots,n\}$ such that $F \cup G = \{1,\ldots,n\}$ and $F \cap G = \phi$. And let F and G be the working set and inactive set in the active set algorithm respectively. Considering the following column partition of the matrix $X = [X_F, X_G]$ where $X_F \in \mathbb{R}^{m\times|F|}$, $X_G \in \mathbb{R}^{m\times|G|}$, and $|F|$, $|G|$ are the number of F and G respectively, we can rewrite (9) as:

$$\begin{bmatrix} \alpha_F \\ \alpha_G \end{bmatrix} = \begin{bmatrix} X_F^TX_F & X_F^TX_G \\ X_G^TX_F & X_G^TX_G \end{bmatrix} \begin{bmatrix} \beta_F \\ \beta_G \end{bmatrix} - \begin{bmatrix} X_F^Ty \\ X_G^Ty \end{bmatrix} + \tfrac{\lambda}{2}$$

where $\beta_F,\ \alpha_F \in \mathbb{R}^{|F|}$, $\beta_G,\ \alpha_G \in \mathbb{R}^{|G|}$, $\beta = (\beta_F, \beta_G)$ and $\alpha = (\alpha_F, \alpha_G)$. Then we can compute values of the variables β_F and α_G by the following iterative procedure:

$$\min_{\beta_F \in R^{|F|}} ||X_F\beta_F - y||_2^2 + \lambda \sum_{i\in F} \beta_i \quad (10)$$

$$\alpha_G = X_G^T(X_F\beta_F - y) + \tfrac{\lambda}{2} \quad (11)$$

And the optimal solution is given by $\beta = (\beta_F, 0)$ and $\alpha = (0, \alpha_G)$. Algorithm 1 summarizes the optimal procedure. As suggested in sparse code algorithms for computer vision and pattern recognition [23], we also normalize each column of the dataset X to have unit l^2-norm. One merit of this normalization step is to easily tune the l^1 regularization item λ.

3.2. Efficient Nonnegative Sparse Coding Algorithm

In Algorithm 1, α_i controls the working set F. In each iteration, the index r corresponding to the minimum a_r is added to the working set F. Looking at Eq.(11), there are three parts in α_i. The first two parts of α_i are $x_i^TX_F\beta_F$ and x_i^Ty respectively. Here, we denote $X_F\beta_F$ by $\hat{y}$. Proposition 1 shows the relationship between the value of x_i^Ty and the value of the l^2 distance $||x_i - y||_2$. If $||x_i||_2^2 = 1$, $||x_j||_2^2 = 1$, and $x_i^Ty \geq x_j^Ty$, x_j will be far away from y than x_i. Based on Proposition 1, we categorize the relationship between the value of α_i and the distance from x_i to y into four cases in Table 1.

Proposition 1 *For* $\forall\ x_i,\ x_j$*, and* y*, if* $||x_i||_2^2 = 1$*,* $||x_j||_2^2 = 1$*, and* $x_i^Ty \geq x_j^Ty$*, then the inequality* $||x_i - y||_2 \leq ||x_j - y||_2$ *holds true.*
Proof sketch. *Given that* $||x_i||_2^2 = 1$*,* $||x_j||_2^2 = 1$ *, and* $x_i^Ty \geq x_j^Ty$*, we have* $(x_i^Tx_i - 2x_i^Ty + y^Ty) \leq (x_j^Tx_j - 2x_j^Ty + y^Ty)$*. Hence,* $||x_i - y||_2^2 \leq ||x_j - y||_2^2$.

For the case 1 and case 4 in Table 1, the λ in Algorithm 1 plays a role of a truncation function. Considering the inequality $a_r < 0$ in the step 4 of Algorithm and $\lambda > 0$, the inequality $x_r^T(X_F\beta_F - y) + \frac{\lambda}{2} < 0$ can be written as $x_r^T(X_F\beta_F - y) < -\frac{\lambda}{2}$. This means that although there may be a sample x_i corresponding large α_i value ($\alpha_i < 0$) to further reduce the objective, the regularization item λ restricts this sample from the working set F. In the case 1 and case 4, we learn if x_i is nearer to y than x_j, the α_i will be smaller than α_j. Hence the λ plays a role of a truncation function and always removes faraway samples ($-\frac{\lambda}{2} \leq \alpha_i < 0$).

For the case 2 and case 3 in Table 1, the λ in Algorithm 1 plays a role of a discrimination function. Looking at Eq.(11), there are two items to decide the value of α_i. The sample x_i that is close to the test sample y may have a large value α_i so that it does not satisfy the inequality in step 4. However, the α_j with respect to a faraway sample x_j can also have a small value. If a sample is redundant in the data set X, it will be potentially restricted from the working set F. Hence the λ potentially makes Algorithm 1 compute a discriminate code.

Combining the four cases, we propose an efficient nonnegative sparse coding algorithm in Algorithm 2 based on the *cluster assumption* which states that two samples are likely to have the same class label if they are connected by a path. Under the cluster assumption, the nonnegative coefficient β_i with respect to x_i actually plays as a clustering indicator. In Algorithm 1, we firstly consider the case 1

[2] Since the solution β is assumed to be sparse, we can use LCP to efficiently find a sparse active set.

Table 1. The relationship between the value of α_i and the distance from x_i to y

	$-x_i y$	$x_i^T \hat{y}$ ($\hat{y} \doteq X_F \beta_F$)	$\alpha_i = x_i^T \hat{y} - x_i^T y$
Case 1	$\|\|x_i - y\|\|_2 \leq \|\|x_j - y\|\|_2$	$\|\|x_i - \hat{y}\|\|_2 \geq \|\|x_j - \hat{y}\|\|_2$	$\alpha_i \leq \alpha_j$
Case 2	$\|\|x_i - y\|\|_2 \leq \|\|x_j - y\|\|_2$	$\|\|x_i - \hat{y}\|\|_2 \leq \|\|x_j - \hat{y}\|\|_2$	$\alpha_i \leq \alpha_j$ or $\alpha_i \geq \alpha_j$
Case 3	$\|\|x_i - y\|\|_2 \geq \|\|x_j - y\|\|_2$	$\|\|x_i - \hat{y}\|\|_2 \geq \|\|x_j - \hat{y}\|\|_2$	$\alpha_i \leq \alpha_j$ or $\alpha_i \geq \alpha_j$
Case 4	$\|\|x_i - y\|\|_2 \geq \|\|x_j - y\|\|_2$	$\|\|x_i - \hat{y}\|\|_2 \leq \|\|x_j - \hat{y}\|\|_2$	$\alpha_i \geq \alpha_j$

and case 2 and compute a nonnegative sparse code from the nearest data set in step 5. We make use of the nearest neighbor parameter n_{knn} instead of the regularization λ. Secondly, based on the clustering assumption, we assume that a test sample can only be expressed by its relative clusterings. Hence we compute an informative and discriminate sparse code only from its relative clusterings in step 6.

Algorithm 2 Efficient Nonnegative Sparse Coding Algorithm

```
1: **Input:** data matrix $X = [X_1, X_2, ..., X_c] \in \mathbb{R}^{m \times n}$
   for $c$ classes, a test sample $y \in \mathbb{R}^{m \times 1}$, the number of
   nearest neighbor $n_{knn}$ ($n_{knn} \leq min(n, d)$).
2: **Output:** sparse code $\beta$.
3: Normalize the columns of $X$ and $y$ to have unit $l^2$-norm.
4: Compute a nearest subset $I^1$ in $X$ to $y$ according to the
   nearest neighbor criterion, and set $X^1 = \{x_i | i \in I^1\}$.
5: Solve the nonnegative least squares problem:
       $\beta^* = \arg\min_{\beta} ||X^1 \beta - y||_2^2 \; s.t. \; \beta \geq 0$      (12)
6: Set $I^2 = \{i | \beta_i > 0 \; and \; i \in I^1\}$ and $X^2 = \{X_c | \; x_i \in X_c \; and \; i \in I^2\}$,
   solve the $l^1$ regularized nonnegative least squares problem:
       $\beta^* = \arg\min_{\beta} ||X^2 \beta - y||_2^2 + \lambda ||\beta||_1 \; s.t. \; \beta \geq 0$      (13)
7: Set $\beta = \mathbf{0} \in \mathbb{R}^{n \times 1}$ and then set $\beta_{I^2} = \beta^*$.
```

4. Sparse Probability Graph for Discriminative Semi-supervised Learning

The basic assumption behind the graph-based machine learning methods is the *cluster assumption* [21]. Under the cluster assumption, positive weights of the graph $\mathscr{G}$ with respect to x_i actually play as a clustering indicator. Based on the l^1-graph [22], we further consider the following sparse probability model on the dataset $X_{\hat{i}}$:

$$\min ||w_i||_0 \quad s.t. \quad X_{\hat{i}} w_i = x_i \quad and \quad w_{ij} \geq 0 \qquad (14)$$

where $w_i \in \mathbb{R}^{(n-1) \times 1}$ and $X_{\hat{i}} w_i = x_i$ is an underdetermined linear system. The optimization problem in Eq.(14) is NP-hard [7, 3]. Fortunately, if the solution w_i is sparse enough, it is unique and global [3]. We can find this solution by replacing the l^0-norm by an l^1-norm [7, 3]. Then we have

$$\min ||x_i - X_{\hat{i}} w_i||_2^2 + \lambda ||w_i||_1 \quad s.t. \quad w_{ij} \geq 0 \qquad (15)$$

The non-negative sparse vector w_i with respect to x_i deduced in Eq.(15) characterizes how the rest samples contribute to the sparse representation of x_i so that it can essentially recover the clustering relation among samples. Moreover, the non-negative sparse representation in Eq.(15) naturally indicates the adjacency structure of the data. Consequently the adjacency structure and weights are determined simultaneously.

After all of the non-negative sparse weights are computed, we can construct a sparse matrix W by

$$W(i,j) = \begin{cases} w_{ij} / \sum\limits_{j'=1}^{n-1} w_{ij'} & j < i \\ w_{i(j-i)} / \sum\limits_{j'=1}^{n-1} w_{ij'} & j > i \end{cases} \qquad (16)$$

Since we expect that the values of the weights only reflect the relationship of the data, we normalize the weights in Eq.(16) to make them represent the probability. Intuitively, this W can be treated as the weight matrix of the graph $\mathscr{G}$. The $W(i,j)$ directly reflects how similar datum x_i is to datum x_j. Since $\sum_{j=1}^{n} W(i,j) = 1$, the $W(i,j)$ can also be explained as the probability that the x_j and x_i belong to the same cluster. Hence we denoted the graph $\mathscr{G}$ based on Eq.(16) by sparse probability graph (SPG).

A key step in SPG is to compute the nonnegative sparse code in Eq.(15). Algorithm 2 is used to compute the sparse code. Considering that W is a sparse matrix, we can make use of the label propagation algorithm [21] to propagate the labels of unlabeled samples. For discriminative semi-supervised learning, the first l samples $X_L = \{x_i\}_{i=1}^{l}$ are labeled and the remaining samples $X_U = \{x_u\}_{i=l+1}^{n}$ are unlabeled in the c-class dataset X. Let M be a set of $n \times c$ matrices with nonnegative real-valued entries. Any matrix $F \in M$ corresponds to a specific classification on X that labels x_i as $\arg\max_{j \leq c} F_{ij}$. Let T be a $n \times c$ matrix where $T_{ij} = 1$ if x_i is labeled as the j-th class, and $T_{ij} = 0$ otherwise, and for unlabeled samples, $T_{uj=0} (1 \leq j \leq c)$.

Algorithm 3 Sparse Probability Graph for Discriminative Semi-supervised Learning

```
Input: $X = \{x_1, \dots, x_l, x_{l+1}, \dots, x_n\}$, $\{x_i\}_{i=1}^{l}$ are labeled and $\{x_u\}_{i=l+1}^{n}$ are unlabeled, the initial label matrix $T$, the number of the nearest neighbors $n_{knn}$, and the constant $\lambda$.
Normalize the columns of $X$ to have unit $l^2$-norm.
for $i = 1$ to $n$ do
    Compute a nearest subset $I^1$ in $X_{\hat{i}}$ to $y$ according to the nearest neighbor criterion, and set $X^1 = \{x_j | j \in I^1\}$.
    Solve the nonnegative least squares problem:
        $w_i^* = \arg\min_{w_i} ||X^1 w_i - y||_2^2 \quad s.t.\ w_i \geq 0 \qquad (17)$
    Set $w_i = \mathbf{0} \in \mathbb{R}^{(n-1)\times 1}$ and then set $w_{iI^1} = w_i^*$.
end for
Construct the probability matrix $W$ according to Eq.(16) and set $F_0 = T$. And iterate $F_{t+1} = p_1 W F_t + (1 - p_1)T$ until convergence.
According to the label $F_{t+1}$, divide $X$ into $[X_1, X_2, \dots, X_c]$.
for $i = 1$ to $n$ do
    Set $I^2 = \{j | w_{ij} > 0\}$ and $X^2 = \{X_c | x_j \in X_c\ and\ j \in I^2 and\ j \neq i\}$, solve the following $l^1$ regularized nonnegative least squares problem:
        $w_i^* = \arg\min_{w_i} ||X^2 \beta - y||_2^2 + \lambda ||\beta||_1 \quad s.t.\ w_i \geq 0 \quad (18)$
    Set $w_i = \mathbf{0} \in \mathbb{R}^{(n-1)\times 1}$ and then set $w_{iI^2} = w_i^*$.
end for
Construct the probability matrix $W$ according to Eq.(16) and set $F_0 = T$. And iterate $F_{t+1} = p_1 W F_t + (1 - p_1)T$ until convergence.
Output: $F^* = F_{t+1}$.
```

Algorithm 3 summarizes the algorithm procedure of sparse probability graph for discriminative semi-supervised learning. First, we solve a nonnegative least squares problem for each sample on its nearest neighbor subset. Second, we construct a coarse SPG and make use of the label propagation algorithm to predict the labels. Since $\sum_{j=1}^{n} W(i,j) = 1$ and $0 < p_1 < 1$, the sequence $\{F_{t+1}\}$ in the label propagation algorithm will converge to $(1 - p_1)(I - p_1 W)^{-1} Y$ [21]. Third, based on the coarsely predicted labels, we compute informative and discriminate nonnegative sparse codes to construct a fine SPG. Last, the labels of unlabeled samples are propagated on the fine SPG.

The computation cost of Algorithm 3 mainly involves three parts: the nearest neighbor sorting, sparse code and label propagation. The computation complexity of the nearest neighbors sorting is $O(nlog(n))$ [3]. The computation costs of computing sparse codes are only relative to n_{knn} and $|I^2|$ (the number of I^2). Suppose that $n_{knn} << n$ and $|I^2| << n$, the computation costs of computing sparse codes can be neglected. The label propagation step requires $O(n^2)$ to compute matrix multiplication. As a result, the computation cost of Algorithm 3 is $O(nlog(n) + n^2)$.

5. Experimental Verification

To evaluate the proposed sparse probability graph algorithm, we compare it with the knn-graph algorithm, the linear label propagation algorithm and the l^1-graph algorithm on non-negative data, real data and sparse data. All algorithms are implemented in MATLAB on an AMD Quad-Core 2.30GHz Windows XP machines with 2GB memory.

5.1. Experimental setting

Database. To fairly compare different methods, we select four different data sets which are from different machine learning applications and consist of different formats of features. As suggested by [23], we normalize the samples to have unit norm. Descriptions of the four data sets are as follows.

ORL Database[4]: The ORL Database contains ten different images each of 40 distinct subjects. The images were taken at different times, varying the lighting, facial expressions and facial details. Each image is manually cropped and normalized to the size of 32 × 32 pixels.

UCI ISOLET Dataset[5]: The ISOLET data set is used to predict which letter-name was spoken. The features include spectral coefficients, contour features, sonorant features, pre-sonorant features, and post-sonorant features. The dimension of feature is 617 and the number of samples is 1559.

TDT2 Document Database[6]: The TDT2 corpus consists of 11,201 on-topic documents which are classified into 96 semantic categories. We use the top 9 categories for our experimental evaluation. Each document is represented as a normalized term-frequency vector, with top 500 words selected according to mutual information. For each category, 60 documents are randomly selected for training. Note that the feature vector in this dataset is sparse.

COIL Database [17]: The Columbia Object Image Library (COIL-100) is a database of 7,200 color images of 100 objects (72 images per object). All images are resized to 32 × 32. For each object, the odd images are selected for training. Hence, there are 3200 images in training set.

Algorithm Setting. The details of compared techniques are:

1) *knn-graph*: The knn-graph is conducted with two configurations as far as classification error rates. In ***knn-graph0***, the number of the nearest neighbor is set to 4; and

[3] If we make use of C++ Language to implement the sorting operation, we can sort one million 3000-dimension data samples per second on one personal computer.

[4] http://www.face-rec.org/databases/

[5] http://www.ics.uci.edu/mlearn/MLRepository.html

[6] http://www.nist.gov/speech/tests/tdt/tdt98/index.htm

Table 2. Comparison classification error rates for semi-supervised algorithms: classification error rate (%) ± standard deviation. The bold numbers are the lowest error rates for each configuration and the percentage number after the data set name is the percentage of the labeled samples [24].

Dataset	knn Graph1	knn Graph2	LLP	SPG1	SPG2	l^1-graph0	l^1-graph1	l^1-graph2
ORL(50%)	15.7±3.5	22.7±3.2	9.6±2.7	7.9 ± 2.6	**7.7** ± 2.6	9.6 ± 2.6	7.8 ± 2.4	8.6 ± 2.7
ORL(60%)	13.2±3.1	20.6±4.1	7.6±2.2	5.8 ± 2.2	**5.6** ± 2.3	6.5 ± 2.0	5.7 ± 2.6	5.8 ± 2.3
ORL(80%)	9.4±3.1	16.4±4.7	4.3±2.1	3.0 ± 1.9	**2.9** ± 2.0	4.1 ± 2.3	3.1 ± 1.7	3.4 ± 2.2
ISOLET(50%)	20.1±1.3	18.3±1.9	14.0±1.0	**13.9**±1.2	14.1±1.3	21.5 ± 1.9	15.4 ± 1.4	15.1 ± 1.3
ISOLET(60%)	19.0±1.7	16.3±1.5	11.8±1.6	11.2±1.5	**11.0**±1.6	17.3 ± 1.7	12.2 ± 1.5	11.9 ± 1.3
ISOLET(80%)	15.0±2.3	13.8±2.1	8.4±1.7	**7.5**±1.4	7.6±1.8	10.9 ± 2.1	7.8 ± 1.6	7.9 ± 1.7
TDT2(50%)	21.3±2.5	16.8±2.3	15.3±1.7	13.8±2.3	**13.5**±1.9	41.7 ± 6.1	34.6 ± 3.1	-
TDT2(60%)	20.3±2.2	15.6±2.4	13.7±2.2	12.1±1.9	**11.9**±1.8	35.5 ± 4.2	32.9 ± 3.1	-
TDT2(80%)	19.3±2.7	14.6±3.7	12.1±2.5	**10.2**±3.2	10.2±2.7	26.8 ± 4.0	28.9 ± 3.8	-
COIL(50%)	16.8±0.8	24.3±1.0	12.1±0.7	10.9±0.7	**10.7**±1.1	16.6±0.9	-	-
COIL(60%)	15.3±1.0	22.1±1.1	10.0±0.9	9.6±0.8	**9.4**±0.8	14.8±0.9	-	-
COIL(80%)	13.2±1.3	19.0±1.6	8.2±0.9	7.9±1.0	**7.7**±1.0	11.9±1.3	-	-

in ***knn-graph1***, the number of the nearest neighbor is set to 10. The heat kernel parameter in heat kernel [1] is well tuned on each dataset to achieve the best results.

2) *LLP*: we follow the lines of linear label propagation [21] to construct a graph. The neighborhood size in LLP is set to 40 to achieve the best results.

3) l^1*-graph*: we compare its three models, which are different in the aspects of robustness and computational strategy [23]. l^1***-graph0***: The graph weights are computed by Eq.(2) via an active set algorithm based on [15]. [7] The l^1 regularized item λ is empirically set to 0.005 to achieve the best results. l^1***-graph1***: The graph weights are computed by Eq.(3) via a primal-dual algorithm for linear programming based on [2, 4].[8] The setting of the algorithm is the same as the one in [23]. l^1***-graph2***: The graph weights are computed by Eq.(4) via an active set algorithm based on [15].[9] The λ is empirically set to 0.005 to achieve the best results.

4) *SPG*: we implement its two models. In ***SPG1***, the graph weights in (15) are computed by Algorithm 1 . The l^1 regularized item λ is empirically set to 0.001. We denote the Algorithm 3 by ***SPG2***. The n_{knn} is set to one-quarter of the size of the training set and the λ is set to 0.001.

5.2. Numerical Results

We follow the approaches in [24][22][5] to quantitatively evaluate different graph based semi-supervised learning methods. All the experiments were repeated 50 times; the means and standard deviations of these 50 runs are reported. In each run, a number of data samples are randomly labeled.

Table 2 tabulates the classification error rates for semi-supervised algorithms. We observe from the numerical results that:

1) in most cases, the SPG1 or SPG2 based semi-supervised learning algorithm achieves the lowest error rates as compared to semi-supervised algorithms. Since the SPG algorithms can determine the adjacent structure of the data and graph probability weights automatically and simultaneously, they can predict the labels of unlabeled samples more accurately. Although two SPG algorithms compute the same objective function, their classification error rates are different due to different optimization strategies. Experimental results demonstrate that the approximate strategy in Algorithm 3 is good enough for discriminative semi-supervised learning.

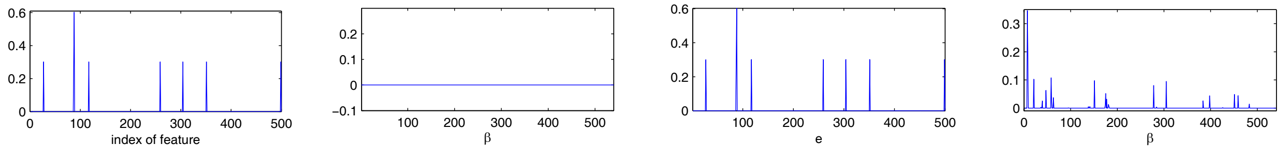
2) l^1-graph1 based semi-supervised learning algorithm consistently outperforms semi-supervised algorithms based on l^1-graph0 and l^1-graph2. Although three l^1 methods are based on the same objective, they make use of different strategies to optimize it. As a result, their accuracy is different. We consider that those results are coincident with the results reported in [25]. Since the features in TDT2 data set are also sparse, the l^1 optimization models used in the l^1-graph fail to yield a correct sparse code. Hence the l^1-graph based semi-supervised learning algorithms obtain high classification error rates.

3) For the ORL dataset and the COIL dataset, the knn Graph can achieve classification error rates under a small value of the number of the nearest neighbor. But for the ISOLET dataset and the TDT2 dataset, the knn

[7] http://redwood.berkeley.edu/ bruno/sparsenet/

[8] http://www.acm.caltech.edu/l1magic/

[9] http://redwood.berkeley.edu/bruno/sparsenet/



(a) (b) (c) (d)

Figure 1. (a) A data sample in TDT2 data set. (b) Sparse coefficient β computed by model (3). (c) Noise item e computed by model (3) corresponding to (b). (d) Sparse code β computed by model (7).

Graph can achieve classification error rates under a large value of the number of the nearest neighbor. It seems that the number of the nearest neighbor of knn-graph depends on the dataset. As illustrated in Fig. 2, all sparse codes based methods can select an adaptive neighborhood, which potentially make them outperform knn-graph.

5.3. Sparsity

Both sparse code and nonnegative sparse code can yield a sparse representation for machine learning tasks. However, to our best knowledge, there still have no works to discuss the sparse characteristics of the two methods and to investigate how sparse is enough for machine learning tasks. Fig. 2 shows the sparsity of different methods measured by the number of nonzeros in the coefficient β. We learn that the number of nonzeros of the two SPGs are smaller than those of the two l^1-graphs. It seems that the nonnegative sparse code based algorithms can learn a more sparse code than sparse code based algorithms. As illustrated in Table 2, this nonnegative sparse code is informative and discriminative enough for machine learning tasks. Fig. 2 also validates that the proposed SPG can learn a sparse and adaptive neighborhood graph for graph-based discriminative semi-supervised learning.

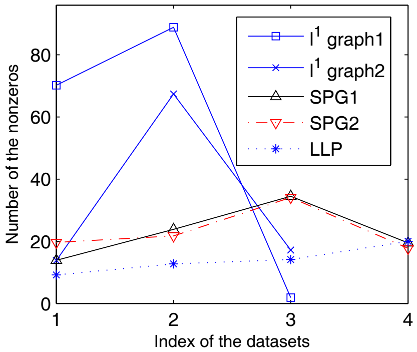


Figure 2. Number of the nonzeros (Sparsity) of different methods. The x-axis indicates the indices of the four data sets.

Since the neighborhood size of LLP is set to 40, LLP learns the most sparsest solution. However, this sparse solution is computed on the nearest neighborhood dataset so that it may lose global information of the whole dataset. As a result, LPP cannot achieve the lowest error rate as compared with the sparse representation based methods.

Fig. 1 (a) shows an interesting observation in TDT2 data set. We see that the feature of TDT2 data is sparse. Fig. 1 (b) and (c) show the values of coefficient β and noise item e computed by model (3) respectively. Since the l^1 optimization model in Eq.(3) treats the coefficient and noise item equally in the l^1 objective, the l^1 optimization algorithm estimates all entries of the sparse data as noise. The sparse coefficient contains no information. As a result, l^1-graph based semi-supervised algorithm fails. Fig. 1 (d) further shows the values of coefficient β computed by model (7). The non-negative sparse representation algorithm can still yield a sparse representation such that SPG based semi-supervised algorithms achieve small error rate.

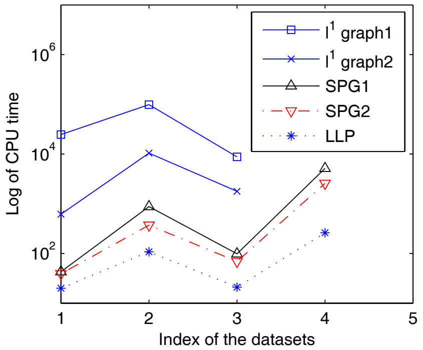


Figure 3. Computational costs of different methods. The x-axis indicates the indices of the four data sets.

5.4. Computational Cost

Computation complexity of construction of a graph is an important issue for semi-supervised learning. Fig. 3 shows the overall CPU time of different methods on all four data sets. Although l^1-graph1 is more informative than l^1-graph2, the computational costs of l^1-graph1 are extremely large, especially when the number of samples is large or the dimension is high. Compared with the l^1-graphs, the computational costs of the SPGs are very slim. On the ISOLET dataset, the CPU time of SPG1 is 874s whereas the CPU time of SGP2 is only 372s. On the COIL dataset, the CPU time of SPG1 is 5140s whereas the CPU time of SPG2 is 2575s. Since SPG2 compute the nonnegative sparse code on a small dataset so that it can save half of the CPU time as compared to SPG1.

6. Conclusion

This paper proposes a new graph model, named sparse probability graph (SPG), for graph-based machine learning methods. The weights of the SPG are derived by nonnegative sparse codes which naturally play as clustering indicators and reflect the importance of a sample for reconstructing a given pattern. To compute the nonnegative sparse weights of SPG, efficient nonnegative sparse algorithms are proposed based on the l^0-l^1 equivalence theory. The new algorithms result in a speedup for nonnegative sparse coding, allowing us to learn larger sparse codes against l^1 graph. Extensive experimental results demonstrate that the nonnegative sparse codings are informative and discriminative for graph based semi-supervised learning.

Acknowledgment

This work was supported in part by the Research Foundation for the Doctoral Program of the Ministry of Education of China (#20100041120009), 985 Project in Sun Yat-sen University (# 35000-3181305), NSF-Guangdong (#U0835005), and the Natural Science of Foundation of China (#61075051, #60971095).

References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[2] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[3] A. M. Bruckstein, M. Elad, and M. Zibulevsky. On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations. *IEEE Transactions on Information Theory*, 54(11):4813–4820, 2008.

[4] E. Candes and J. Romberg. *l1-magic: recovery of sparse signals via convex programming*. http://www.acm.caltech.edu/l1magic/, 2005.

[5] B. Cheng, J. Yang, S. Yan, Y. Fu, and T. S. Huang. Learning with ℓ1-graph for image analysis. *IEEE Transactions on Image Processing*, 4:858–866, 2010.

[6] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[7] D. Donoho and J. Tanner. Sparse nonnegative solutions of underdetermined linear equations by linear programming. In *Proceedings of the National Academy of Sciences*, volume 102, pages 9446–9451, 2005.

[8] J. Duchi and Y. Singer. Efficient online and batch learning using forward-backward splitting. *Journal of Machine Learning Research*, 10:2873–2898, 2009.

[9] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] R. He, B. G. Hu, W. S. Zheng, and Y. Q. Guo. Two-stage sparse representation for robust recognition on large-scale database. In *Twenty- Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, 2010.

[11] R. He, W.-S. Zheng, and B.-G. Hu. Maximum correntropy criterion for robust face recognition. *(accepted) IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[12] P. O. Hoyer. Non-negative sparse coding. In *IEEE Workshop on Neural Networks for Signal Processing*, pages 557–565, 2002.

[13] P. O. Hoyer and P. Dayan. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[14] Y. Ji, T. Lin, and H. Zha. Mahalanobis distance based non-negative sparse representation for face recognition. In *Int. Conf. on Machine Learning and Applications (ICMLA)*, 2009.

[15] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Neural Information Processing Systems NIPS*, 2006.

[16] W. Liu, P. P. Pokharel, and J. C. Principe. Correntropy: Properties and applications in non-gaussian signal processing. *IEEE Transactions on Signal Processing*, 55(11):5286–5298, 2007.

[17] S. Nene, S. Nayar, and H. Murase. Columbia object image library (coil-20). *Technical Report CUCS-005-96*, 1996.

[18] L. Portugal, J. Judice, and L. Vicente. A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables. *Mathematics of Computation*, 63(208):625–643, 1994.

[19] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1):267–288, 1996.

[20] N. Vo, B. Moran, and S. Challa. Nonnegative-least-square classifier for face recognition. In *Proceedings of the 6th International Symposium on Neural Networks:Advances in Neural Networks*, pages 449–456, 2009.

[21] F. Wang and C. S. Zhang. Label propagation through linear neighborhoods. *IEEE Trans. on knowledged and data engineering*, 20(1):55–67, 2008.

[22] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of IEEE*, 98(6):1031–1044, 2010.

[23] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[24] S. Yan and H. Wang. Semi-supervised learning by sparse representation. In *SIAM International Conference on Data Mining, SDM*, 2009.

[25] A. Y. Yang, S. S. Sastry, A. Ganesh, and Y. Ma. Fast l_1-minimization algorithms and an application in robust face recognition: A reivew. In *International Conference on Image Processing*, 2010.