Hypergraph Spectral Learning for Multi-label Classification

Liang Sun
Arizona State University
Tempe, AZ 85287
sun.liang@asu.edu

Shuiwang Ji
Arizona State University
Tempe, AZ 85287
shuiwang.ji@asu.edu

Jieping Ye
Arizona State University
Tempe, AZ 85287
jieping.ye@asu.edu

ABSTRACT

A hypergraph is a generalization of the traditional graph in which the edges are arbitrary non-empty subsets of the vertex set. It has been applied successfully to capture high-order relations in various domains. In this paper, we propose a hypergraph spectral learning formulation for multi-label classification, where a hypergraph is constructed to exploit the correlation information among different labels. We show that the proposed formulation leads to an eigenvalue problem, which may be computationally expensive especially for large-scale problems. To reduce the computational cost, we propose an approximate formulation, which is shown to be equivalent to a least squares problem under a mild condition. Based on the approximate formulation, efficient algorithms for solving least squares problems can be applied to scale the formulation to very large data sets. In addition, existing regularization techniques for least squares can be incorporated into the model for improved generalization performance. We have conducted experiments using large-scale benchmark data sets, and experimental results show that the proposed hypergraph spectral learning formulation is effective in capturing the high-order relations in multi-label problems. Results also indicate that the approximate formulation is much more efficient than the original one, while keeping competitive classification performance.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications - Data Mining

General Terms

Algorithm

Keywords

Multi-label classification, hypergraph, spectral learning, least squares, canonical correlation analysis, efficiency, regularization

KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.
Copyright 2008 ACM 978-1-60558-193-4/08/08 ...$5.00.

1. INTRODUCTION

Multi-label learning studies the problem where each instance is associated with a set of labels. Such type of problems occurs in many important applications, such as protein function classification [9], text categorization [25], and semantic scene classification [4]. In contrast to traditional classification tasks where the classes are mutually exclusive, the classes in multi-label learning are overlapped and correlated, which makes it challenging to predict all relevant labels for a given instance. A straightforward method to perform multi-label learning is to divide it into a number of one-against-all binary classification problems. A potential limitation of this approach is that each label is treated independently, and the correlation among different class labels is ignored. Various approaches have been proposed in the past to exploit the correlation information contained in multiple labels [15, 16, 20, 21, 22, 26, 27]. In [9], a multi-label SVM model, called RankSVM, was proposed based on novel definitions of loss and margin for multi-label problems.

In this paper, we propose to employ hypergraph [1] to capture the correlation information among different labels for improved classification performance. A hypergraph is a generalization of the traditional graph in which the edges are arbitrary non-empty subsets of the vertex set. It has been applied for domains where higher-order relations such as co-authorship exist [1, 28]. We propose to employ hypergraphs to exploit the higher-order relations among multiple instances sharing the same label in multi-label learning. Specifically, we propose to construct a hyperedge for each label, and include all instances annotated with a common label into one hyperedge, thus capturing their joint similarity. Following the spectral graph embedding theory [7], we propose to compute the low-dimensional embedding through a linear transformation, which preserves the instance-label relations captured by the hypergraph. The projection is guided by the label information encoded in the hypergraph. In addition, we show the close relationship between the proposed formulation and the well-known dimensionality reduction algorithm, Canonical Correlation Analysis (CCA) [14].

The resulting hypergraph learning formulation amounts to solving an eigenvalue problem, which may be computationally expensive for large-scale problems. Motivated from our key observation obtained from CCA, we propose an approximate hypergraph spectral learning formulation, which can be reformulated as a least squares problem. Following the least squares formulation, different regularization techniques [12] can be employed to improve the generalization performance of the model. Moreover, efficient algorithms for solving least squares can be applied to scale the algorithm to very large data sets [18]. We have conducted experiments using large-scale benchmark data sets, and our results demonstrate the effectiveness of the proposed hypergraph spectral learning formulation for multi-label classification. Results also show that the scalability of the approximate formulation is far superior to that of the original one, while keeping competitive classification performance.

The rest of this paper is organized as follows. We review the basics of hypergraph Laplacian and the least squares in Section 2. We present our multi-label learning formulation based on hypergraph in Section 3. The approximate hypergraph spectral learning formulation based on least squares is presented in Section 4. We report experimental results in Section 5 and the paper concludes in Section 6.

Notations n, d, and k denote the number of training samples, the data dimensionality, and the number of labels, respectively. $x_i \in \mathbb{R}^d$ denotes the ith instance, $y_i \in \mathbb{R}^k$ contains the label information for x_i. $X = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{d\times n}$ represents the data matrix, and $Y = [y_1, y_2, \cdots, y_n] \in \mathbb{R}^{k\times n}$ is the matrix representation for label information. Both $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$ are assumed to be centered, i.e., $\sum_{i=1}^n x_i = 0$, and $\sum_{i=1}^n y_i = 0$. $\mathcal{L} \in \mathbb{R}^{n\times n}$ is the normalized Laplacian matrix for the constructed hypergraph, and $\mathcal{S} = I - \mathcal{L}$ is the similarity matrix, where I is the identity matrix. e is a vector of all ones with an appropriate length.

2. BACKGROUND

We review the basics of hypergraph and the least squares technique in this section.

2.1 Hypergraph

A hypergraph [1] is a generalization of the traditional graph in which the edges, called hyperedges, are arbitrary non-empty subsets of the vertex set. In a hypergraph $G = (V, E)$, V is the vertex set and E is the edge set where each $e \in E$ is a subset of V. The degree of a hyperedge e, denoted as $\delta(e)$, is the number of vertices in e. The degree of every edge in a traditional graph is 2, and it is therefore called a "2-graph". The degree of a vertex $v \in V$, $d(v)$, is defined as

$$d(v) = \sum_{v\in e, e\in E} w(e),$$

where $w(e)$ is the weight associated with the hyperedge $e \in E$. The diagonal matrix forms for $\delta(e)$, $d(v)$, $w(e)$ are denoted as D_e, D_v, W_H, respectively. The vertex-edge incidence matrix $J \in \mathbb{R}^{|V|\times|E|}$ is defined as

$$J(v, e) = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We can use the higher-order relations encoded in a hypergraph for multi-label learning. The Laplacian matrix from a traditional graph is widely used for learning from graphs [6]. Regarding the hypergraph, one can either define hypergraph Laplacian directly or expand it into a 2-graph. It has been shown that the Laplacians defined in both ways are similar [1]. In addition, the eigenvectors of these Laplacians have been shown to be useful for learning higher-order relations, and that there is a close relationship between their hypergraph Laplacians and the structure of the hypergraph. In this section, we briefly review two commonly used expansions as well as one hypergraph Laplacian.

2.1.1 *Clique Expansion*

In clique expansion, each hyperedge is expanded into a clique. Denote by $G_c = (V_c, E_c)$ the 2-graph expanded from hypergraph $G = (V, E)$ using the clique expansion. We have $V_c = V$ and $E_c = \{(u, v) : u \in e, v \in e, e \in E\}$. The edge weight $w_c(u, v)$ of G_c is given by

$$w_c(u, v) = \sum_{u,v\in e, e\in E} w(e). \tag{2}$$

In the expanded 2-graph, the vertex degree denoted by $d_c(u)$ can be obtained readily, and the corresponding diagonal matrix form is denoted as D_c. Then the normalized Laplacian of G_c is given by $\mathcal{L}_c = I - \mathcal{S}_c$, where S_c is defined as

$$\mathcal{S}_c = D_c^{-1/2} J W_H J^T D_c^{-1/2}. \tag{3}$$

Intuitively in clique expansion, the similarity between two vertices is proportional to the weights of common labels, thus capturing the instance-label relations.

2.1.2 *Star Expansion*

In star expansion, a new vertex is introduced for each hyperedge and this new vertex is connected to each vertex in this hyperedge. Specifically, for a hypergraph $G = (V, E)$, the vertex and edge sets of the star-expanded 2-graph, denoted as V_* and E_*, are defined as $V_* = V \cup E$ and $E_* = \{(u, e) : u \in e, e \in E\}$, respectively. Thus each hyperedge in G is expanded into a star in G_*, which is a bipartite graph.

The weight $w_*(u, e)$ of an edge (u, e) in G_* is given by

$$w_*(u, e) = w(e)/\delta(e). \tag{4}$$

Since $V_* = V \cup E$, we can assume that in V_*, all $v \in V$ are ordered before $e \in E$. Let $M \in \mathbb{R}^{|V|\times|E|}$ denote the weight between the vertices constructed from V and the vertices from E in G_*. The adjacency matrix for G_* can be obtained readily from M. Based on the adjacency matrix, the degree for all vertices can be computed. We use D_{*v} and D_{*e} to denote the diagonal matrices of vertex degrees for vertices in V and E in the expanded graph $G_* = (V_*, E_*)$, respectively.

Suppose $x^T = [x_v^T, x_e^T]$ is an eigenvector of Laplacian $\mathcal{L}_*$ with corresponding eigenvalue γ, where $\mathcal{L}_*$ is the normalized Laplacian for the expanded graph, it can be verified that

$$BB^T x_v = (\gamma - 1)^2 x_v, \tag{5}$$

where $B = (D_*^v)^{-1/2} M (D_*^e)^{-1/2}$. In spectral learning, the eigenpair of the Laplacian $\mathcal{L}_*$ is essential for learning. Recall that only x_v corresponds to the real vertices in the original hypergraph while x_e corresponds to the artificial vertices, thus x_v and its corresponding eigenvalue are more important for learning. It follows from Eq. (5) that we can obtain the eigenpair (x_v, γ) from BB^T and simplify the computation. We denote $S_* = BB^T$ in the following discussion, and the equivalent matrix form is

$$\mathcal{S}_* \;=\; D_{*v}{}^{-1/2} M D_{*e}{}^{-1} M^T D_{*v}{}^{-1/2}. \tag{6}$$

Intuitively in star expansion, the artificial vertices associated with the labels connect to the vertices from the corresponding label, thus capturing the interaction between the data points and the labels. The relationship among different labels can then be captured by their interactions with the data points. More details can be found in [1, 29].

2.1.3 Hypergraph Laplacian

Several other methods define the "hypergraph Laplacian" using analogies from the graph Laplacian [19, 28]. Following the random walk model, Zhou et al. [28] proposed the following normalized hypergraph Laplacian $\mathcal{L}_z$:

$$\mathcal{L}_z = I - \mathcal{S}_z, \tag{7}$$

where

$$\mathcal{S}_z = I - L_z = D_v^{-\frac{1}{2}} J W_H D_e^{-1} J^T D_v^{-\frac{1}{2}}.$$

In the random walk model, given the current position $u \in V$, the walker first chooses a hyperedge e over all hyperedges incident with u with the probability proportional to $w(e)$, and then chooses a vertex $v \in e$ uniformly at random. Note that each label corresponds to a hyperedge in our framework. The transition probability between the nodes associated with two labels captures their similarity. More details can be found in [28].

2.2 Least Squares

Least squares is a classical technique for both regression and classification [3]. Given a set of observations $x_i \in \mathbb{R}^d (1 \leqslant i \leqslant n)$ and their corresponding targets $t_i \in \mathbb{R}^k (1 \leqslant i \leqslant n)$, the goal is to fit a linear model

$$t = W^T x + b$$

to the data, where $b \in \mathbb{R}^k$ is the bias vector, and $W \in \mathbb{R}^{d \times k}$ is the weight matrix. Assuming that both the observations $\{x_i\}_{i=1}^n$ and the targets $\{t_i\}_{i=1}^n$ are centered, the bias term becomes zero and can be ignored. The optimal W can be computed by minimizing the following sum-of-squares error function:

$$\sum_{i=1}^{n} \|W^T x_i - t_i\|_2^2 = \|W^T X - T\|_F^2, \tag{8}$$

where $T = [t_1, \cdots, t_n] \in \mathbb{R}^{k \times n}$. The optimal matrix W is

$$W_{LS} = (XX^T)^\dagger X T^T, \tag{9}$$

where † denotes the pseudo-inverse. When least squares is applied for classification, T is known as the class indicator matrix. In binary-class problem we can define $t_i \in \{-1, 1\}$. For multi-class or multi-label problems, a similar definition can be applied for each label separately [12, 24].

Least squares problems can be solved efficiently using existing techniques, which is very attractive for large-scale data mining. Many iterative algorithms, including the conjugate gradient method, have been proposed in the literature [10, 18]. Compared with the direct methods, these iterative algorithms converge very fast for large and sparse problems.

3. MULTI-LABEL LEARNING WITH HYPERGRAPH

In this section, we present our multi-label learning formulation based on hypergraph. We also show the close relationship between the proposed formulation and Canonical Correlation Analysis (CCA).

3.1 Hypergraph Spectral Learning

We propose to learn from multi-label data by exploiting the spectral property of hypergraph that encodes the correlation information among labels. To capture the correlation among labels, we create a hyperedge for each label and include all data points relevant to this label into the hyperedge. Based on the Laplacian of the constructed hypergraph, we propose the *hypergraph spectral learning* framework for learning a low-dimensional embedding through a linear transformation W which solves the following optimization problem:

$$\begin{aligned} \min_W \quad & \operatorname{trace}(W^T X \mathcal{L} X^T W) \\ \text{subject to} \quad & W^T X X^T W = I_k, \end{aligned} \tag{10}$$

where $\mathcal{L}$ is the normalized Laplacian of the hypergraph, and $W \in \mathbb{R}^{d \times k}$ is the projection matrix. In this formulation, the objective function in Eq. (10) attempts to preserve the inherent relationship among data points captured by the Laplacian [2]. It follows from the traditional spectral graph embedding theory [7]. Intuitively, data points that share many common labels tend to be close to each other in the embedded space.

Define the matrix $\mathcal{S}$ as

$$\mathcal{S} = I - \mathcal{L}. \tag{11}$$

It follows from the property of the normalized Laplacian $\mathcal{L}$ that $\mathcal{S} \in \mathbb{R}^{n \times n}$ reflects the normalized similarities between different instances (or vertices). Hence, the original optimization problem in Eq. (10) can be reformulated equivalently into the following form:

$$\begin{aligned} \max_W \quad & \operatorname{trace}(W^T X \mathcal{S} X^T W) \\ \text{subject to} \quad & W^T X X^T W = I_k. \end{aligned} \tag{12}$$

It can be verified that the optimal W^* to Eq. (12) consists of the eigenvectors corresponding to the largest eigenvalues for the following eigenvalue problem:

$$\left((XX^T)^\dagger (X \mathcal{S} X^T) \right) w = \gamma w, \tag{13}$$

where $\gamma \in \mathbb{R}$ denotes the eigenvalue.

To avoid the singularity of XX^T, a regularization term is commonly added. This leads to the following *regularized hypergraph spectral learning* problem:

$$\left((XX^T + \lambda I)^{-1} (X \mathcal{S} X^T) \right) w = \gamma w. \tag{14}$$

Recall from Section 2 that different Laplacians can be constructed from the hypergraph that encodes the label information, resulting in different spectral learning algorithms.

3.2 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) [14] is a well-known technique for finding the correlation between two sets of variables. In CCA, two different views of the same set of objects are given, and a projection is computed for each view such that their correlation is maximized in the dimensionality-reduced space. CCA is commonly used for supervised dimensionality reduction in which one of the view is derived from the data, and another view is derived from labels. In this setting, the data can be projected into a lower-dimensional space directed by the label information.

We consider the case in which the dimensionality reduction for $X \in \mathbb{R}^{d \times n}$ is directed by the label information encoded in $Y \in \mathbb{R}^{k \times n}$, and $y_i \in \mathbb{R}^k$ contains the label information for $x_i \in \mathbb{R}^n$. When YY^T is nonsingular, it can be verified that the multiple projection vectors for X under certain orthogonality constraints can be computed simultaneously by solving the following optimization problem [11]:

$$\begin{array}{ll} \max\limits_{W} & \text{trace}(W^TXY^T(YY^T)^{-1}YX^TW) \\ \text{subject to} & W^TXX^TW = I_\ell, \end{array} \tag{15}$$

where $W \in \mathbb{R}^{d\times\ell}$ is the projection matrix for X and ℓ is the number of projection vectors. It is clear that CCA is a special case of the formulation in Eq. (12) in which the similarity matrix $\mathcal{S}$ is given by

$$\mathcal{S} = Y^T(YY^T)^{-1}Y.$$

4. APPROXIMATE HYPERGRAPH SPECTRAL LEARNING

The spectral learning formulation in Eq. (12) involves an eigenvalue problem in Eq. (13), which may be computationally expensive for large-scale problems. In this section, we propose an approximate hypergraph spectral learning formulation for improved efficiency. Our experimental results show that this approximate formulation is much more efficient and scalable than the original one, while keeping competitive classification performance.

One first key observation is that when we use star expansion, clique expansion, or Zhou's formulation to derive the Laplacian matrix, the similarity matrices in Eq. (11), denoted as $\mathcal{S}_c$, $\mathcal{S}_*$, and $\mathcal{S}_z$, respectively, are all positive semi-definite. It is clear that the similarity matrix derived from CCA, denoted as $\mathcal{S}_{cca}$ is also positive semi-definite. Mathematically, they all can be decomposed into the following form:

$$\mathcal{S} = H_0H_0^T, \tag{16}$$

for some matrix $H_0 \in \mathbb{R}^{n\times k}$.

Note that we assume that the training data are centered, i.e., X is of zero mean. Thus, we have $XP = X$, where P is centering matrix defined as $P = I_n - \frac{1}{n}ee^T$. It can be verified that P is symmetric and $PP = P$. It follows that

$$X\mathcal{S}X^T = (XP)\mathcal{S}(P^TX^T) = X\left(P\mathcal{S}P\right)X^T.$$

That is, we can simply assume that $\mathcal{S}$ is centered in terms of both rows and columns. Hence, we have

$$e^T\mathcal{S}e = e^TH_0H_0^Te = 0,$$

and $H_0^Te = 0$. These properties can be easily verified in CCA where $H_0 = Y^T(YY^T)^{-1/2}$. Moreover, in CCA, the matrix H_0 has orthonormal columns, i.e., $H_0^TH_0 = I_k$.

4.1 Approximation Assumption

Following CCA, we propose to approximate the hypergraph Laplacian by requiring the corresponding H_0 to have orthonormal columns. We show that an H_0 with orthonormal columns can simplify the computation considerably. In particular, we show that under this approximation, the optimization problem in Eq. (12) can be reduced to a least problem under a mild condition. We first define the following assumption:

ASSUMPTION 1. *$\mathcal{S}$ can be decomposed into the form:*

$$\mathcal{S} = HH^T, \tag{17}$$

where $H \in \mathbb{R}^{n\times k}$ satisfies the following properties:

$$1. \quad H^TH = I_k, \tag{18}$$

$$2. \quad H^Te = 0. \tag{19}$$

Based on Assumption 1, we show that the hypergraph spectral learning formulation can be solved efficiently. Define two matrices as follows:

$$C_X = XX^T \in \mathbb{R}^{d\times d}, \tag{20}$$

$$C_H = XHH^TX^T \in \mathbb{R}^{d\times d}. \tag{21}$$

It follows that the eigenvalue problem in Eq. (13) can be expressed as

$$C_X^\dagger C_Hw = \gamma w.$$

Let

$$X = U\Sigma V^T = U_1\Sigma_rV_1^T$$

be the Singular Value Decomposition (SVD) of X, where $r = \text{rank}(X)$, U and V are orthogonal matrices, $\Sigma \in \mathbb{R}^{d\times n}$ is diagonal, and $U_1\Sigma_rV_1^T$ is the compact SVD of X, $U_1 \in \mathbb{R}^{d\times r}$, $V_1 \in \mathbb{R}^{n\times r}$, and $\Sigma_r \in \mathbb{R}^{r\times r}$. Define $A \in \mathbb{R}^{r\times k}$ as

$$A = V_1^TH, \tag{22}$$

and let $A = P\Sigma_AQ^T$ be the SVD of A, where $P \in \mathbb{R}^{r\times r}$ and $Q \in \mathbb{R}^{k\times k}$ are orthogonal, and $\Sigma_A \in \mathbb{R}^{r\times k}$ is diagonal. Then the eigendecomposition of $C_X^\dagger C_H$ can be expressed as

$$\begin{aligned} C_X^\dagger C_H &= U_1\Sigma_r^{-2}U_1^TXHH^TX^T \\ &= U_1\Sigma_r^{-1}AA^T\Sigma_rU_1^T \\ &= U\begin{bmatrix} I_r \\ 0 \end{bmatrix}\Sigma_r^{-1}P\Sigma_A\Sigma_A^TP^T\Sigma_r[I_r, 0]U^T \\ &= U\begin{bmatrix} \Sigma_r^{-1}P & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} \Sigma_A\Sigma_A^T & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} P^T\Sigma_r & 0 \\ 0 & I \end{bmatrix}U^T. \end{aligned}$$

Thus, under Assumption 1, the solution to the hypergraph spectral learning formulation is given by the top ℓ eigenvectors of matrix $C_X^\dagger C_H$, which can be expressed as

$$W_{HS} = U_1\Sigma_r^{-1}P_\ell, \tag{23}$$

where P_ℓ contains the first ℓ columns of P.

4.2 Least Squares Formulation

Consider the least squares formulation in Eq. (8) with the class indicator matrix $\tilde{T}$ given by

$$\tilde{T} = H^T, \tag{24}$$

It follows from Eq. (9) that the optimal solution is given by

$$W_{LS} = (XX^T)^\dagger X\tilde{T}^T = U_1\Sigma_r^{-1}P\Sigma_AQ^T. \tag{25}$$

We next show that all diagonal elements of Σ_A are ones if rank$(X) = n - 1$, which is equivalent to the linearly independence condition on the data X before centering.

THEOREM 4.1. *Assume that rank$(X) = n-1$ and $Xe = 0$ (data centered). Then all diagonal elements of Σ_A are ones.*

PROOF. It can be verified that

$$Xe = 0 \Leftrightarrow U_1\Sigma_rV_1^Te = 0 \Leftrightarrow V_1^Te = 0.$$

It follows from $V_1^Te = 0$ and $V_1 \in \mathbb{R}^{n\times(n-1)}$ that $V_{ex} = [V_1, \frac{1}{\sqrt{n}}e] \in \mathbb{R}^{n\times n}$ is an orthogonal matrix. Thus, we have

$$I_n = V_{ex}V_{ex}^T = V_1V_1^T + \frac{1}{n}ee^T.$$

It follows from Eq. (22) that

$$A^TA = H^TV_1V_1^TH = H^T(I_n - \frac{1}{n}ee^T)H$$
$$= H^TH - \frac{1}{n}(H^Te)(e^TH) = I_k. \qquad (26)$$

Thus all nonzero singular values of A in Σ_A are ones. □

Since rank$(\Sigma_A) = k$, there are k nonzero eigenvalues. If we choose $\ell = k$, then

$$W_{HS} = U_1\Sigma_r^{-1}P_k. \qquad (27)$$

Hence, the only difference between W_{LS} and W_{HS} lies in the orthogonal matrix Q^T in W_{LS}. Since orthogonal projections preserve the pairwise Euclidean distance between data points, W_{HS} and W_{LS} are equivalent when classifiers such as the K-Nearest-Neighbor (KNN) based on the Euclidean distance and linear Support Vector Machines (SVM) are applied in the projected space.

4.3 Extensions Using Regularization

The above discussions show that under Assumption 1, the proposed hypergraph spectral learning formulation can be formulated as a least squares problem under a mild condition. Regularization can then be applied to control the model complexity and improve the classification performance. Linear regression using the 2-norm regularization, called ridge regression [13], minimizes the penalized sum-of-squares error function. By using the target matrix $\tilde{T}$ as defined in Eq. (24), we obtain the following 2-norm regularized least squares formulation:

$$\min L_2(W, \lambda) = \|W^TX - \tilde{T}\|_F^2 + \lambda\|W\|_F^2, \qquad (28)$$

where $\lambda > 0$ is the regularization parameter.

4.4 Optimal Approximation

Recall that in CCA, the matrix $\mathcal{S}$ satisfies Assumption 1. Thus, the CCA formulation can be transformed equivalently into least squares problem. However, this assumption does not hold for hypergraphs in general. We propose to approximate $\mathcal{S} = H_0H_0^T$ in Eq. (16) by HH^T subject to the constraint that H satisfies the properties in Assumption 1. This leads to the following optimization problem:

$$\min_{H\in\mathbb{R}^{n\times k}} \quad \|\mathcal{S} - HH^T\|_F^2 \qquad (29)$$
$$\text{subject to} \quad H^TH = I_k,$$
$$H^Te = 0.$$

The solution to this problem is summarized in the following theorem.

Theorem 4.2. *Let $\mathcal{S}$, H, and H_0 be defined as above. Then the optimal H^* that solves the optimization problem in Eq. (29) is given by the top k left singular vectors of H_0.*

Proof. It follows from basic matrix properties that

$$\|\mathcal{S} - HH^T\|_F^2$$
$$= \text{trace}\left((\mathcal{S} - HH^T)^T(\mathcal{S} - HH^T)\right)$$
$$= \text{trace}(\mathcal{S}^T\mathcal{S}) - 2\text{trace}(\mathcal{S}^THH^T) + \text{trace}(HH^THH^T)$$
$$= \text{trace}(\mathcal{S}^T\mathcal{S}) - 2\text{trace}(H^T\mathcal{S}H) + k,$$

where the last equality follows since $H^TH = I_k$. Hence, the minimization problem in Eq. (29) is equivalent to the maximization of trace$(H^T\mathcal{S}H)$, which can be considered as a special case of the more general optimization problem in [8]. The optimal H is given by the eigenvectors of $\mathcal{S}$ corresponding to the top k eigenvalues. From Eq. (16), $\mathcal{S} = H_0H_0^T$, thus the optimal H^* is given by the top k left singular vectors of H_0. Also note that $H_0^Te = 0$, which leads to ${H^*}^Te = 0$. □

4.5 Efficient Implementation

Recall that our proposed spectral learning formulation in Eq. (10) involves an eigenvalue problem in Eq. (13), which has a time complexity of $O(n^2d)$ assuming that $n < d$. Using efficient algorithms for solving sparse least squares problem, we show that the approximate formulation has a much lower time complexity. In particular, the approximate formulation involves the following two steps for computing the projection matrix W:

1. Compute the top k left singular vectors of H_0 for H.
2. Solve a least square problem using H^T as the class indicator matrix.

The complexity of the first step is $O(nk^2)$. In the second step, we solve k least squares problems. In our implementation, we use the LSQR algorithm proposed in [18], which is an implementation of a conjugate gradient type method for solving sparse least squares problems [5]. Note that the original matrix $X \in \mathbb{R}^{d\times n}$ may be sparse in many applications such as text document processing. However, after centering, X is not sparse any more. In order to keep the sparsity of X, the vector x_i is augmented by an additional component as $\tilde{x}_i^T = [1, x_i^T]$. This new component acts as the bias for least squares. The extended X is denoted as $\tilde{X} \in \mathbb{R}^{(d+1)\times n}$, and the revised least squares problem is formulated as

$$\min_{\tilde{W}} \|\tilde{W}^T\tilde{X} - \tilde{T}\|_F^2, \qquad (30)$$

where $\tilde{W} \in \mathbb{R}^{(d+1)\times k}$. For a new data point $x \in \mathbb{R}^d$, its projection is given by $\tilde{W}^T[1; x]$.

The computational cost of each iteration of LSQR is $O(3n + 5d + 2dn)$ [18]. Since the least squares problems are solved k times, the overall cost of LSQR is $O(Nk(3n + 5d + 2dn))$, where N is the total number of iterations.

When the matrix $\tilde{X}$ is sparse, the cost is notably reduced. Suppose the number of nonzero elements in $\tilde{X}$ is z. The overall cost of LSQR is reduced to $O(Nk(3n + 5d + 2z))$. In summary, the total time complexity for solving the approximate formulation via LSQR is

$$O(nk^2 + Nk(3n + 5d + 2z)).$$

We observe in our experiments that the iterative algorithm converges quickly, and a total of $N = 100$ iterations is enough in most cases.

5. EXPERIMENTAL EVALUATION

In this section, we empirically evaluate the effectiveness and efficiency of the proposed formulations.

5.1 Experimental Setup

We use both low-dimensional and high-dimensional data sets in the experiments, including the scene data set [4], the

Table 1: Summary of statistics of the data sets. n is number of data points, d is the dimensionality, and k is the number of labels.

Data Set	n	d	k
Scene	2407	294	6
Yeast	2417	103	14
Yahoo\Arts&Humanities	3712	23146	26
Yahoo\Computers&Internet	6270	34096	33

yeast data set [9], and two high-dimensional document data sets [16, 21]. The scene and yeast data sets are available at http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets, and the two document data sets from Yahoo! are available at http://www.kecl.ntt.co.jp/as/members/ueda/yahoo.tar.gz. For all data sets, the labels that contain less than 50 instances are removed. We follow the feature selection methods studied in [23] for text documents and extract different numbers of terms to investigate the performance of algorithms. The statistics of these data sets are summarized in Table 1.

For each data set, a transformation matrix W is learned from the training set, and it is then used to project the test data onto a lower-dimensional space in which the linear Support Vector Machine (SVM) is applied for each label separately. The Receiver Operating Characteristic (ROC) score is employed to evaluate the classification performance. All experiments are performed on a PC with Intel Core 2 Duo T7200 2.0G CPU and 2G RAM and all algorithms in our experiments are implemented in Matlab.

We denote the hypergraph spectral learning formulation in Eq. (13) and its regularized version in Eq. (14) as HG and HG^{λ}, respectively. The approximate (least squares) hypergraph spectral learning formulation and its regularized version are denoted as $lsHG$ and $lsHG^{\lambda}$, respectively. The subscripts c, $*$, z, and cca are used to denote the formulations derived from clique expansion, star expansion, Zhou's formulation, and CCA, respectively.

5.2 Performance Comparison

We compare the performance of the hypergraph spectral learning formulation and its approximate least squares formulation on the scene and yeast data sets. For each data set, we increase the size of the training set gradually from 100 to 900 with a step size around 100, and generate ten random training/test partitions in each case. Figures 1 and 2 plot the averaged performance of the hypergraph spectral learning formulation and the least squares formulation on the scene and yeast data sets for all definitions of Laplacian. We can observe that the performance of both methods for all definitions of Laplacian is very close. This shows that the proposed approximate least squares formulation is close to the original one.

We also compare the regularized hypergraph spectral learning formulation with its approximate counterpart. In this experiment, the size of training set is fixed at 900, and different values of the regularization parameter λ are used. We report the results on the scene and document data sets in Figures 3 and 4. It can be observed that the performance of the regularized algorithms perform much better than their counterparts without regularization. Results also show that the performance of the regularized hypergraph spectral learning formulation and that of its approximate counterpart are similar in all cases. This again justifies the use of the approximate formulations proposed in this paper.

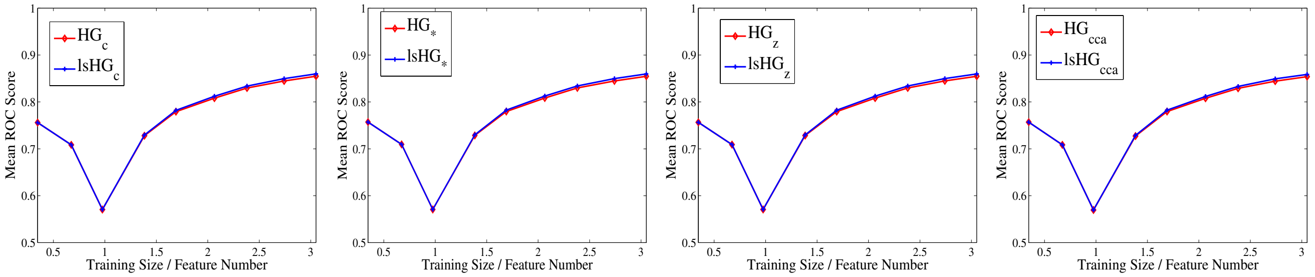
Table 2: Summary of mean ROC scores over all labels for all compared algorithms. All the features of the scene and yeast data sets are used while the most frequent 2000 terms of the Arts and Computers data sets are used.

Algorithm	Scene	Yeast	Arts	Computers
Training size	900	900	2000	2000
HG_c	0.8548	0.6460	0.5215	0.5369
HG_c^{λ}	0.9216	0.6496	**0.8106**	0.7968
$lsHG_c$	0.8598	0.6493	0.5213	0.5375
$lsHG_c^{\lambda}$	**0.9255**	0.6523	0.8130	0.7996
HG_*	0.8546	0.6529	0.5209	0.5384
HG_*^{λ}	0.9216	0.6558	0.8051	0.7970
$lsHG_*$	0.8597	0.6540	0.5209	0.5381
$lsHG_*^{\lambda}$	**0.9255**	0.6565	0.8067	0.7999
HG_z	0.8547	0.6518	0.5206	0.5389
HG_z^{λ}	0.9216	0.6521	0.8074	0.7991
$lsHG_z$	0.8597	0.6537	0.5205	0.5389
$lsHG_z^{\lambda}$	0.9254	0.6559	0.8087	**0.8025**
HG_{cca}	0.8538	0.6547	0.5213	0.5341
HG_{cca}^{λ}	0.9215	0.6555	0.8015	0.7917
$lsHG_{cca}$	0.8586	0.6561	0.5215	0.5340
$lsHG_{cca}^{\lambda}$	**0.9255**	**0.6568**	0.8036	0.7952
RankSVM	0.9001	0.6294	0.8073	0.7893

To evaluate the relative performance of all variants of the hypergraph spectral learning formulation and those of the approximate formulation comprehensively, we report the mean ROC scores over all labels and all splittings for each algorithm in Table 2. The performance of RankSVM [9] is also reported. 3-fold cross validation is used to estimate the optimal regularization parameters. We also apply SVM to each label independently and our results show that it is less effective than RankSVM and is thus omitted in the table. We can observe from Table 2 that the performance of the hypergraph spectral learning formulations is close to that of the approximate formulations. Moreover, regularization algorithms always outperform those without regularization, which justifies the use of regularization. All of the proposed methods with regularization perform better than the RankSVM algorithm. Results also show that three different similarity matrices S lead to the similar performance, which is consistent with the theoretical analysis in [1].

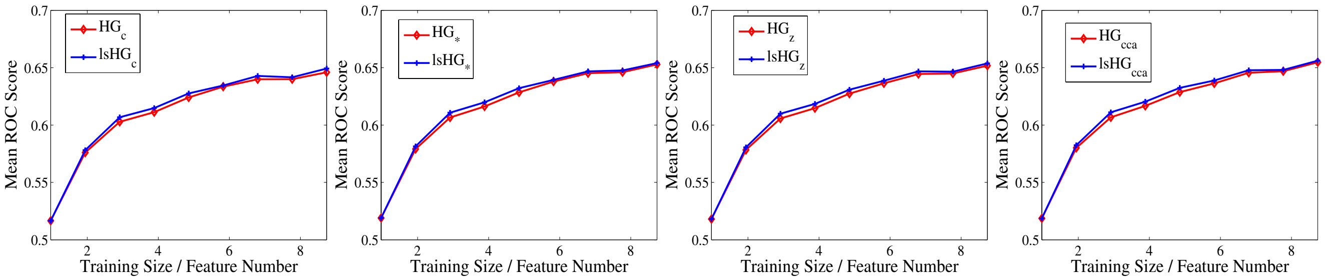
5.3 Scalability Comparison

In this subsection, we study the scalability of regularized hypergraph spectral learning formulation and its approximate counterpart as the sample size and data dimensionality increase. The approximate formulation is solved by the LSQR algorithm [18]. Figure 5 depicts the computation time of the two formulations on the Yahoo\Arts&Humanities data set as the data dimensionality increases with the training size fixed at 2000. It can be observed that the computation time for both algorithms increases steadily as the data dimensionality increases. The computation time of the approximate formulation is substantially less than that of the hypergraph spectral learning formulation. In fact, the com-



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 1: Comparison of the hypergraph spectral learning formulation (HG) and the approximate fromulation ($lsHG$) on the scene data set. The x-axis is the ratio of the training set size to the data dimensionality, and the y-axis is the mean ROC score. The training set size varies from 100 to 900 with a step size around 100.



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 2: Comparison of the hypergraph spectral learning formulation (HG) and the approximate fromulation ($lsHG$) on the yeast data set. The x-axis is the ratio of the training set size to the data dimensionality, and the y-axis is the mean ROC score. The training set size varies from 100 to 900 with a step size around 100.

putation time of the approximate formulation is less than 0.5 second for all tested data dimensionality. We also evaluate the scalability of the formulations in terms of the training sample size. Figure 6 plots the computation time of the regularized hypergraph spectral learning and the regularized approximate formulations on the Yahoo\Arts&Humanities data set when the training sample size increases gradually with the data dimensionality fixed at 2000. We can observe that the approximate formulation is much more scalable than the original formulation when the sample size increases. We have not further increased the training size in this experiment, due to the high computational cost of the original formulation.

To investigate the scalability of the approximate formulations further, we use the entire Yahoo\Computers&Interne data set which contains 6270 samples with 34096 dimensions. We increase the data dimensionality gradually with the training size fixed at 6270 and the computation time of all four approximate formulations is plotted in Figure 7. We can observe that the proposed approximate formulations take less than 9 seconds in all cases. Our experimental results show that the proposed approximate formulations scale to large data sets, while keeping competitive classification performance as shown in Table 2.

6. CONCLUSIONS AND FUTURE WORK

We present a hypergraph spectral learning formulation for multi-label classification in this paper. In this formulation, a hypergraph is constructed to capture the correlation information contained in different labels. We show that the proposed formulation is reduced to an eigenvalue problem which may be computationally expensive for large-scale problems. We further propose an approximate formulation which amounts to solving a least squares problem. Efficient algorithms for solving least squares problem such as LSQR can thus be employed. Experimental results show that the proposed formulation is effective in capturing the correlation information for multi-label classification. Results also show that the approximate formulation is much more efficient, while keeping competitive classification performance.

Experimental results show that the approximate formulation is very close to the original formulation in terms of classification performance. We plan to perform a theoretical analysis on the approximate formulation in terms of the error bounds of the approximation formulation. The main focus of this paper is on linear models. We plan to extend the hypergraph spectral learning formulation to the kernel-induced feature space, and apply the kernel formulation to multi-label applications such as protein function prediction [17] where only kernels are available.

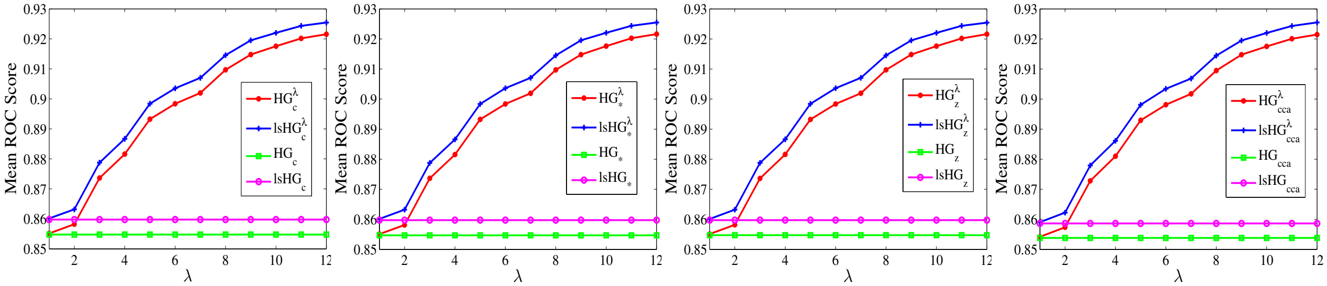
Acknowledgments

This research is sponsored in part by the Arizona State University and by the National Science Foundation Grant IIS-0612069.

7. REFERENCES

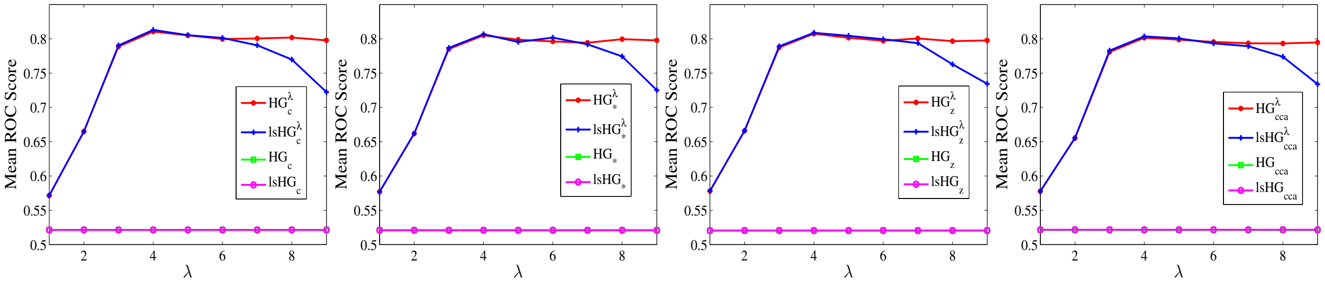
[1] S. Agarwal, K. Branson, and S. Belongie. Higher order learning with graphs. In *ICML*, pages 17–24, 2006.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in*



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 3: Comparison of four methods (HG, $lsHG$, HG^λ, $lsHG^\lambda$) in terms of mean ROC score on the scene data set. The x-axis denotes the index for λ, which is $[0.1, 1, 10, 20, 50, 75, 100, 200, 350, 500, 750, 1000]$, and the y-axis is the mean ROC score. The traninig size is 900. Ten random splittings of the training set and test set are generated, and the averaged performance is reported.



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 4: Comparison of four methods (HG, $lsHG$, HG^λ, $lsHG^\lambda$) in terms of mean ROC score on the Yahoo\Arts&Humanities data set. The x-axis is the index for λ, which is $[1e-06, 1e-4, 0.01, 0.1, 1, 3, 10, 50, 200]$, and the y-axis is the mean ROC score. The traininig size $n = 2000$, and the data dimensionality $d = 2000$.

Neural Information Processing Systems 14, pages 585–591, 2001.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[4] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[5] D. Cai, X. He, and J. Han. SRDA: An efficient algorithm for large-scale discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):1–12, 2008.

[6] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning.* MIT Press, Cambridge, MA, 2006.

[7] F. R. K. Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

[8] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1999.

[9] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems 14*, pages 681–687, 2001.

[10] G. H. Golub and C. F. V. Loan. *Matrix Computations.* Johns Hopkins Press, Baltimore, MD, 1996.

[11] D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16(12):2639–2664, 2004.

[12] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning.* Springer, New York, NY, 2001.

[13] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[14] H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:312–377, 1936.

[15] F. Kang, R. Jin, and R. Sukthankar. Correlated label propagation with application to multi-label learning. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1719–1726, 2006.

[16] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda. Maximal margin labeling for multi-topic text categorization. In *Advances in Neural Information Processing Systems 17*, pages 649–656. 2005.

[17] W. Noble. Support vector machine applications in computational biology. *Kernel Methods in Computational Biology. B. Schoelkopf, K. Tsuda and J.-P. Vert, ed. MIT Press*, pages 71–92, 2004.

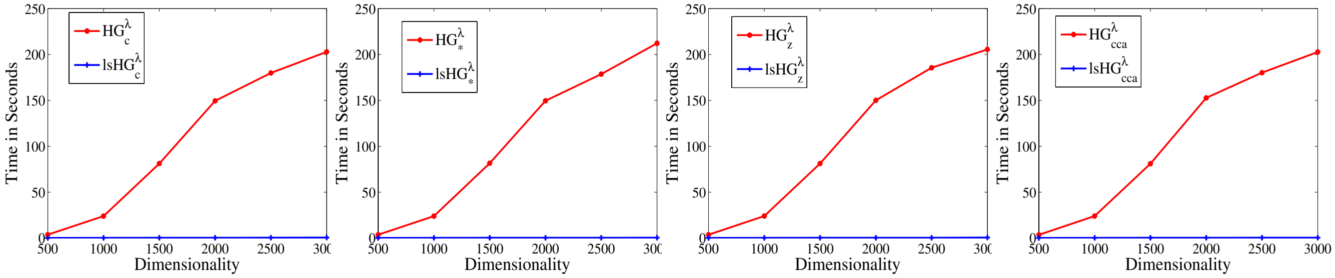
[18] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, 1982.

[19] J. A. Rodríguez. On the laplacian spectrum and walk-regular hypergraphs. *Linear and Multilinear Algebra*, (1):285–297, 2003.

[20] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

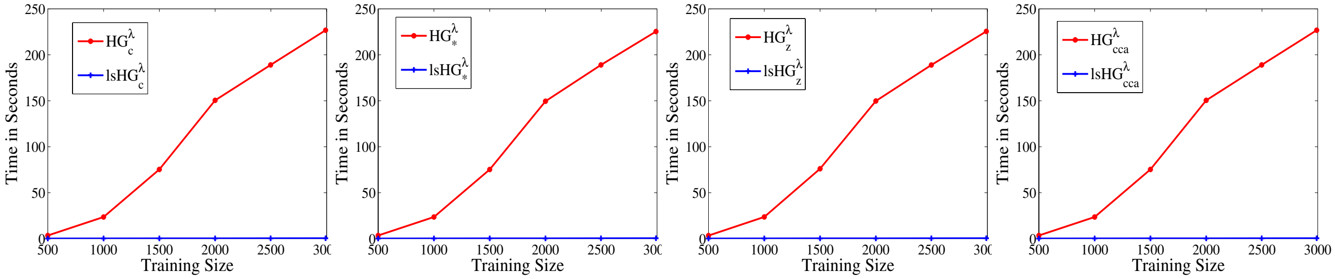
[21] N. Ueda and K. Saito. Parametric mixture models for multi-labeled text. In *Advances in Neural Information Processing Systems 16*, pages 721–728. 2003.

[22] R. Yan, J. Tesic, and J. R. Smith. Model-shared subspace boosting for multi-label classification. In *Proceedings of the thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 834–843, 2007.



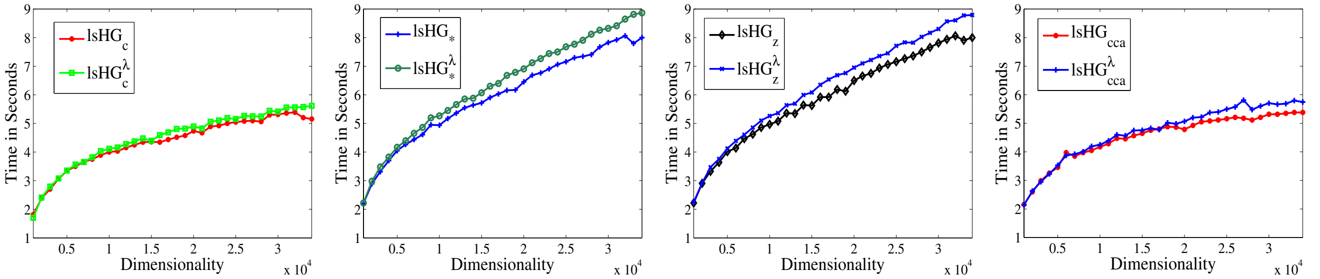
(E) Clique Expansion (F) Star Expansion (G) Zhou's Laplacian (H) CCA

Figure 5: Computation time (in seconds) of the hypergraph spectral learning formulation and its approximate formulation on the Yahoo\Arts&Humanities data set as the data dimensionality increases. The x-axis is the data dimensionality and the y-axis is the computation time. The data dimensionality increases from 500 to 3000 with a step size of 500, and the size of training set is fixed at 2000. The running time of the approximate formulations is much smaller (close to the zero line) in all cases.



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 6: Computation time (in seconds) of the hypergraph spectral learning formulation and its approximate formulation on the Yahoo\Arts&Humanities data set as the size of training set increases. The x-axis is the training size and the y-axis is the computation time. The size of training set increases from 500 to 3000 with a step size of 500, and the data dimensionality is fixed at 2000. The running time of the approximate formulations is much smaller (close to the zero line) in all cases.



(A) Clique Expansion (B) Star Expansion (C) Zhou's Laplacian (D) CCA

Figure 7: Computation time (in seconds) of the approximate formulation with and without regularization on the Yahoo\Computers&Internet data set as the data dimensionality increases. The data dimensionality increases from 1000 to 34,000 with a step size of 1000, and the size of the training set is fixed at 6270.

[23] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, pages 412–420, 1997.

[24] J. Ye. Least squares linear discriminant analysis. In *ICML*, pages 1087–1094, 2007.

[25] S. Yu, K. Yu, V. Tresp, and H.-P. Kriegel. Multi-output regularized feature projection. *IEEE Transactions on Knowledge and Data Engineering*, 18(12):1600–1613, 2006.

[26] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

[27] M.-L. Zhang and Z.-H. Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.

[28] D. Zhou, J. Huang, and B. Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems 19*, pages 1601–1608, 2007.

[29] J. Zien, M. Schlag, and P. Chan. Multilevel spectral hypergraph partitioning with arbitrary vertex sizes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(9):1389–1399, 1999.