Labelset Anchored Subspace Ensemble (LASE) for Multi-label Annotation

Tianyi Zhou  
Centre for Quantum Comput. & Intelligent Syst.  
Faculty of Engineering & IT  
University of Technology Sydney, Australia  
tianyi.zhou@student.uts.edu.au

Dacheng Tao  
Centre for Quantum Comput. & Intelligent Syst.  
Faculty of Engineering & IT  
University of Technology Sydney, Australia  
dacheng.tao@uts.edu.au

ABSTRACT

In multimedia retrieval, multi-label annotation for image, text and video is challenging and attracts rapidly growing interests in past decades. The main crux of multi-label annotation lies on 1) how to reduce the model complexity when the label space expands exponentially with the increase of the number of labels; and 2) how to leverage the label correlations which have broadly believed useful for boosting annotation performance. In this paper, we propose "labelsets anchored subspace ensemble (LASE)" to solve both problems in an efficient scheme, whose training is a regularized matrix decomposition and prediction is an inference of group sparse representations. In order to shrink the label space, we firstly introduce "label distilling" extracting the frequent labelsets to replace the original labels. In the training stage, the data matrix is decomposed as the sum of several low-rank matrices and a sparse residual via a randomized optimization, where each low-rank part defines a feature subspace mapped by a labelset. A manifold regularization is applied to map the labelset geometry to the geometry of the obtained subspaces. In the prediction stage, the group sparse representation of a new sample on the subspace ensemble is estimated by group lasso. The selected subspaces indicate the labelsets that the sample should be annotated with. Experiments on several benchmark datasets of texts, images, web data and videos validate the appealing performance of LASE in multi-label annotation.

Categories and Subject Descriptors

H.3.3 [Information Systems]: Information storage and retrieval

General Terms

Theory

Keywords

Multi-label annotation, Distilled labelsets, Geometry preserving, Subspace ensemble, Low-rank, Group sparse

1. INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ICMR '12, June 5-8, Hong Kong, China  
Copyright ©2012 ACM 978-1-4503-1329-2/12/06 ...$10.00.

In multimedia information retrieval, annotating [3][18] given texts, web data, music, images or videos with concepts or keywords plays a key role in browsing, ranking, indexing, search and navigation. Recently, the rapid rise of data volumes makes the efficiency and effectiveness of manual annotation unacceptable, and triggers a growing need for automatical annotation techniques, which are mainly developed from machine learning methods. In general, each annotation concept or keyword is treated as an independent class and the annotation problem is formulated as a series of binary classification tasks, each of which distinguishes the instance belonging to one class from the others. Such kind of multi-class classification problems can be addressed by various existing deterministic or probabilistic models.

However, an instance is always annotated by multiple concepts in practical multimedia data. Moreover, the real applications often require to annotate an instance with several keywords [16] in order to provide an informative sketch of the object. Hence, the traditional multi-class formulation of annotation faces significant limitations when applying to real problems. Although the multi-label annotation introduced above can be decomposed into several binary classification problems if treating each concept as an independent class, the correlations between different concepts, which have been broadly proved substantial for boosting the annotation performance, are entirely ignored in this case. Another problem of formulating the multi-label annotation as multiple classifications is that the serious imbalance between positive samples and negative samples often results in an unreliable classification result.

In this paper, we aim at developing a novel multi-label learning method for multimedia annotation. Multi-label learning [15][10][5][12] predicts multiple labels that characterize an instance from a set of possible labels. This is exactly the case in annotation problem. Conventional multi-label learning methods seek to find a mapping from the feature space \( \mathcal{X} \subseteq \mathbb{R}^p \) to the label space \( \mathcal{Y} \subseteq \{0, 1\}^k \), wherein \( k \) is the number of labels and \( y_i \equiv 1 \), \( y \in \mathcal{Y} \) denotes annotating label \( i \) to the instance \( x \). Binary relevance (BR) [12] and label powerset (LP) [12] are two natural approaches that construct the mapping by training independent binary classifiers. In particular, BR treats each label as a class, while LP treat each unique \( k \)-dimensional label vector as a class. Both BR and LP do not fully explore the key properties of multi-label data, because BR ignores the label correlations, and LP makes the training samples of each class far less than the prerequisite.

In the multi-label annotation, a central problem is that the label space \( \mathcal{Y} \) will exponentially grow with the increase of labels. Without considering the label structures, the sample complexity for learning the mapping from \( \mathcal{X} \) to \( \mathcal{Y} \) will significantly increase in this case. Moreover, the prediction requires a search in a label space \( \mathcal{Y} \) of size \( 2^k \), which leads to expensive computation. BR
independently predicts each dimension of $y$ in isolation and thus does not encounter this problem with the price of ignoring the label correlations. LP attempts to distinguish each element in $Y$ from the other $2^k - 1$ ones. Thus the size of the training set, which is large enough for binary classification, will be insufficient for LP. This problem also leads to a rapid growth of the model complexity, which increases the training costs. By viewing the problem from the perspective of probabilistic approaches, the exponential growth of $Y$ drastically enlarges the parameter space for modeling $P(y|x)$, $x \in X$, which makes ML intractable in computation.

Another important problem in real applications is that, for a given training set $\{X,Y\}$, the annotation instances in $Y$ often scatter sparsely within the ambient space $Y$. It is therefore difficult to study the structure of $Y$ and reduce its dimensionality.

Most recent multi-label learning approaches investigate the label correlations (or dependencies) to build a structured classification model. They partially solve the first problem by reducing the size of the search space $Y$. For example, the random k-labelsets (RAkEL) method [14] randomly selects an ensemble of subsets from the original labelsets (the set of labels one instance belongs to), and then LP is applied to each subset. The final prediction is obtained by ranking and thresholding of the results on the subsets. Hierarchical binary relevance (HBR) [1] builds a general-to-specific tree structure of labels, where a sample with a label must be associated with its parent labels. A binary classifier is trained on each non-root label. The classifier chain (CC) [11] adopts a greedy method to predict unknown label from features and predicted labels by using a binary classifier. However, the size of the search space in these approaches is much larger than $Y$, and their model complexities are too high to be applied in practice. Also, the instances in $Y$ are always insufficient to generate a reliable estimation of the label space structure, and thus weaken the effectiveness of the structured classification models used in these approaches.

Beyond reducing the problem to several classifications, some other existing learning methods have been reformulated for solving multi-label annotation. For example, the C&W procedure [2] separates multi-label prediction into two stages, i.e., BR and correction of the BR results by leveraging the label dependency. Multi-label dimensionality reduction via dependence maximization (MDDM) [21] maximizes the dependence between feature space and label space, and provides a data preprocessing for other multi-label learning methods. However, these methods cannot provide an explicit modeling of the label correlations and thus their performance improvements due to exploring label structure are limited. Moreover, they bring extra time costs to the training process, so the efficiency is weakened.

### 1.1 Main contributions

In this paper, we consider the ML problem for multi-label annotation by taking an inverse viewpoint: instead of investigating the mapping from $X$ to $Y$, we study the mapping from $Y$ to $X$. To be specific, we expect to map each basis of the label space $Y$ to $X$ in the form of an explainable feature subspace. Fixing the mapping and given an arbitrary instance $x$ with its associated label vector $y$, the feature vector $x$ exists in the ensemble of the subspaces which are mapped by the bases used to represent $y$. Since the label vector is restricted to be 0-1 valued, the bases for the label space $Y$ should also be 0-1 valued. In addition, the 0-1 bases can be interpreted as the labelsets frequently appearing among the label vectors, where each labelset includes the labels those are often simultaneously tagged to one instance. By viewing the problem “labelset distilling (LD)” to extract the bases from the label vectors of the training set according to their distribution in $Y$. An important fact about the obtained 0-1 bases is that they are usually not independent to each other. Thus the correlations between the bases should be preserved after the mapping, i.e., the subspaces in the ensemble are not independent but related to each other by following the correlations between bases. Therefore, learning the $Y$ to $X$ mapping should not only minimize the error of representing instance $y$ in the subspace ensemble mapped by its labelsets, but also maintain the geometry of labelsets in their mapped subspace ensemble.

In establishing the mapping from $Y$ to $X$ for real data, an easily ignored fact is that each given instance cannot be fully depicted by the tagged labels, and thus cannot be completely represented by the subspaces associated with the labelsets we consider in the problem. So the method learning the subspace ensemble should be able to separate the parts that can be explained by the considered labelsets and the parts that cannot. The method also needs to separate the parts explained by different labelsets. Henceforth, the ensemble of subspaces can be directly obtained from the separated parts for different labelsets.

Fixing the subspace ensemble and given a new instance, its labels are predicted by estimating its group sparse representation in the subspace ensemble and combine the labelsets associated with the nonzero entries. There are at most $k$ subspaces that are demanded to be learned (in the case of treating each single label as a labelset), so the model and sample complexity could be substantially smaller than existing methods. The prediction is accomplished by searching in the subspace ensemble, and thus avoids the estimation of the label space structure from insufficient training data.

We develop “labelsets anchored subspace ensemble (LASE)” to find the $Y$ to $X$ mapping that satisfy all the above requirements.

In the training stage, we develop a randomized decomposition of the training data $X$, where $X$ is factorized to the sum of several low-rank parts and a sparse residual. Each of the low-rank part defines the subspace mapped by a particular labelset, while the sparse residual stores the part that cannot be explained by the considered labels. In order to preserve the geometry of labelsets in their mappings, a manifold regularization term constructed from the graph Laplacian of labelsets is imposed to the decomposition. The decomposition is fast due to an application of the bilateral random projection (BRP) based low-rank approximation [22][17]. Its convergence to a stationary point is guaranteed. In the prediction stage of LASE, group lasso estimates the group sparse representation of a given instance in the subspace ensemble, and the nonzero entries indicate the predicted labelsets. The experiments on several benchmark datasets for multi-label annotation imply the competitive effectiveness and efficiency of LASE in multimedia retrieval.

### 2. LASE MODEL

An illustration of LASE mode is given in Figure 1. Granted a sample $x \in X \subseteq \mathbb{R}^p$ and its label vector $y \in Y \subseteq \{0,1\}^k$, fixing the frequently appearing $d$ labelsets as the rows of $D$ (here $D$ can be viewed as a 0-1 dictionary for $y$), if $y$ is represented by a set of labelsets $D_{i \in \Omega}$ selected from the rows of $D$ such that

$$y = \bigcup_{i \in \Omega} D_i = uD, u_{i \in \Omega} = 1, u_{i \notin \Omega} = 0,$$

(1)

then we assume that $x$ can be decomposed as the sum of several components $l$ corresponding to $\Omega$ and a sparse residual $s$

$$x = \sum_{i \in \Omega} l_i + s.$$

(2)

The component $l_i$ exists in a subspace $C_i$, which is the mapping of labelset $D_i$ that represents $y$. Thus $l_i$ is the part of the features
in $x$ that can be explained by the group of labels in labelset $D_i$. The sparse residual $s$ is the part of features in $x$ that neither label in $y$ can explain. The model in (2) reveals the general relationship between the feature space and the label space.

**Figure 1: Illustration of LASE model: mapping from a labelset $D_i$ in label space $\mathcal{Y}$ to a subspace $C_i^l$ in feature space $\mathcal{X}$.**

To be more specific about how $l^i$ is explained by the labelset $D_i$, we assume that the component explained by labelset $D_i$ lies in a linear subspace represented by $C_i^l \in \mathbb{R}^{n \times p}$, i.e., $l^i = \beta_{C_i} C_i^l$, wherein $\beta_{C_i}$ stores the representation coefficients corresponding to $C_i$. Thus the model (2) can be equivalently written as

$$x = \sum_{i=1}^{d} \beta_{C_i} C_i^l + s, \beta_{C_i} s_{\Omega} = 0. \tag{3}$$

If we build the subspace ensemble $C = [C_1^l, \ldots, C_d^l]$ mapped by the $d$ labelsets as a dictionary for $x$, the corresponding representation coefficient vector for $x$ is $\beta = [\beta_{C_1}, \ldots, \beta_{C_d}]$. Then the coefficient group $\beta_{C_i}$ corresponding to the labelset that does not appear in $y$ are zeros, so $\beta$ is group sparse, wherein the groups are $G_i, i = 1, \ldots, d$.

In the training stage of LASE, given training set composed of a feature matrix $X \in \mathbb{R}^{n \times p}$ and its label matrix $Y \in \{0, 1\}^{n \times k}$, we learn the dictionaries $D$ for $\mathcal{Y}$ and $C$ for $\mathcal{X}$ respectively, wherein subspace $C_i^l$ is the mapping of labelset $D_i$ in $X$. Firstly, a “labelset distilling (LD)” method is used to decompose matrix $Y$ as $Y = UD$, $U \in \{0, 1\}^{n \times d}$, $D \in \{0, 1\}^{d \times k}$ via iterative clustering and subtractions of the $h$ unique rows in $Y$. That is, we remove the rows in $Y$ that appear more than one time and obtain matrix $L \in \{0, 1\}^{h \times d}$, and then $L$ is decomposed by LD. Since the decomposition of each row in $L$ on the dictionary $D$ is known after LD, the decomposition of $Y$ can be directly attained. The procedure of each iteration in LD can be sketched as:

**Step 1:** Spectral clustering [7][9] or k-means [8] of $L$’s rows.

**Step 2:** For rows in each cluster, if they share a binary pattern, add the pattern to the labelsets $D$ as a new row and subtract it from the rows of the cluster sharing it in $Y$. Accordingly, add a new column to the $U$, wherein the elements corresponding to the rows in the cluster are set to be 1 and the others are set to be 0.

**Step 3:** Remove the all-zero rows in $L$.

The above procedure is iterated until $L$ is empty, i.e., all the rows in $Y$ are fully represented by the atoms in the dictionary $D$. In this paper, we use spectral clustering to group the rows of $L$ in the first step of each iteration, because the number of clusters obtained by spectral clustering can be adaptively determined by a given threshold $\tau$. In particular, we sort $\xi$ eigenvalues of the Graph Laplacian from small to large, and compute the following metric for each one:

$$e_i = \frac{\sum_{j=1}^{q-i} \lambda_j}{\sum_{j=1}^{q} \lambda_j}, i = 2, \cdots, q. \tag{5}$$
4. TRAINING BY RANDOMIZED DECOMPOSITION

In this section, we introduce the training stage of LASE, which approximately decomposes the training data matrix \( X \in \mathbb{R}^{n \times p} \) into \( X = \sum_{i=1}^{d} L_i^r + S \) with a manifold regularization. The mappings of the \( d \) labelsets, i.e., the subspace ensemble, can be directly obtained from \( L_i^r \) with preserved geometry. For matrix \( L_i^r \), the rows corresponding to the instances with labelset \( D_i \) are nonzero, while the other rows are all zero-vectors. The nonzero rows denote the components explained by labelset \( D_i \) in the feature space. We use \( \Omega_i \) to denote the index set \( \{ j : U_{ji} = 1 \} \), and then the sub-matrix composed of the nonzero rows in \( L_i^r \) is \( L_{i\Omega_i}^r \).

In LASE training, the rank of \( L_{i\Omega_i}^r \) is upper bounded, which indicates that all the components explained by labelset \( D_i \) nearly lie in a linear subspace. The matrix \( S \) is the residual that cannot be explained by the given labelsets. In the decomposition, the cardinality of \( S \) is upper bounded, which makes \( S \) sparse. In order to maintain the geometry of labelsets \( D_i \) after mapping them as the subspace ensemble \( C_i \), we impose a manifold regularization to the \( d \) row centers of \( L_{i\Omega_i}^r \), \( i = 1, \ldots, d \), which are the first bases for the \( d \) subspaces \( C_i \) respectively. The normalized graph Laplacian \( W \) is estimated as:

(1) If LD is not applied and the labelsets \( D_i \) are directly selected as the \( k \) normal unit vectors \( D_i = e_i, e_i \in \mathbb{R}^k, e_i \neq 0 \) with associated \( U = Y \), their normalized graph Laplacian can be estimated from the correlations between the columns of \( Y \) as \( W = I - H^{-1/2}GY^{-1/2} \), wherein \( G \) is the similarity graph (e.g., \( k \)-nearest neighbor graph, or fully connected graph) and diagonal matrix \( H \) stores the associated degrees such that \( H_{ii} = \sum_{j=1}^{k} G_{ij} \). For example, if Gaussian similarity is applied, then \( G_{ij} = \exp \left( \frac{\| (Y^T)_i - (Y^T)_j \|^2}{2\sigma^2} \right) \) for a fully connected graph.

(2) If the labelsets \( D_i \) are the frequent binary patterns obtained by LD such that \( Y = UD \), their normalized graph Laplacian can be estimated from \( D_i \) by following \( W = I - H^{-1/2}GH^{-1/2} \) as well, wherein \( G \) is the similarity matrix and \( H \) is the associated degree matrix. The only difference is the similarity graph \( G_{ij} \) is calculated from \( D_i \) and \( D_j \). For example, Gaussian similarity in this case is \( G_{ij} = \exp \left( \frac{\| (D_i - D_j) \|^2}{2\sigma^2} \right) \).

Given the data matrix \( X \), its label matrix \( Y \in \{0,1\}^{n \times k} = UD \) and the associated normalized graph Laplacian \( W \), the rank of \( L_{i\Omega_i}^r \) is upper bounded by \( r^i \) and the cardinality of \( S \) is upper bounded by \( K \), if the \( d \) rows of \( U \) are the \( d \) row centers \( L_{i\Omega_i}^r \) for \( i = 1, \ldots, d \), the decomposition can be written as solving the following constrained minimization problem:

\[
\min_{L_i^r, S} \left\| X - \sum_{i=1}^{d} L_i^r + S \right\|_F^2 + \lambda \text{tr} \left( \Omega_i^T W \Omega_i \right) \quad \text{s.t.} \quad \text{rank} \left( L_{i\Omega_i}^r \right) \leq r^i, \forall i = 1, \ldots, d, \quad \text{card} \left( \Omega_i \right) \leq K.
\]

In (6), each training instance in \( X \) is decomposed as the sum of several components, which respectively correspond to the labelsets which are annotated to the instance. LASE separates these components explained by different labelsets via building the mapping from each labelset to the feature space. For labelset \( D_i \), we obtain its mapping \( C_i \) in \( X \) as the row space of \( L_{i\Omega_i}^r \).

4.1 Alternating minimization

Although the rank constraint to \( L_{i\Omega_i}^r \) and cardinality constraint to \( S \) are not convex, we will show the optimization in (6) can be solved by alternating minimization, and its linear convergence to a stationary point is guaranteed. Before proceeding to the algorithm details, we firstly rewrite the objective function in (6) as:

\[
\min_{L_{i\Omega_i}^r, S} \left\| X - \sum_{i=1}^{d} L_i^r - S \right\|_F^2 + \lambda \text{tr} \left( \Omega_i^T W \Omega_i \right) = \left\| X - \sum_{i=1}^{d} L_i^r - S \right\|_F^2 + \lambda \sum_{i=1}^{d} \sum_{j=1}^{d} W_{ij} \text{tr} \left( \Omega_i^T \Omega_j \right) \left( \Omega_i^T \Omega_j \right)^T.
\]

Then the optimization problem can be decomposed as the following \( d + 1 \) subproblems, wherein the first \( d \) ones update the \( d \) low rank parts \( L_i^r \), and the last one updates the sparse part \( S \).
For each of the first \(d\) subproblems to solve \(L^i_{\Omega_1}\), we immediately set the gradient of the object to zero and obtain

\[
L^i_{\Omega_1} = \left( I + R^{-1} \right) \left( P^i + \frac{1}{2} Q^i \right) = \left( I - \left( |\Omega| + \frac{|\Omega|^2}{\lambda M^2} \right)^{-1} \bar{T} \bar{T}^T \right) \left( P^i + \frac{1}{2} Q^i \right) . \tag{8}
\]

Since the rank of \(L^i_{\Omega_1}\) is upper bounded by \(r^i\), an additional projection to the low-rank manifold is required, i.e., the solution of \(L^i_{\Omega_1}\) is updated as

\[
L^i_{\Omega_1} := \sum_{q=1}^{r^i} \phi_q U_q V_q^T , U \Phi \Psi^T = \operatorname{svd} \left( \left( I - \left( |\Omega| + \frac{|\Omega|^2}{\lambda M^2} \right)^{-1} \bar{T} \bar{T}^T \right) \left( P^i + \frac{1}{2} Q^i \right) \right) . \tag{9}
\]

The above solution is a hard thresholding of singular values and thus is the global solution for the \(i^{th}\) subproblem.

For the last subproblem in (7), we can simply impose a hard thresholding of matrix entries to obtain the global solution. That is, \(S\) is updated from the \(K\) entries with the largest absolute value in \(X - \sum_{j=1}^{d} L_j\), i.e.,

\[
S := \mathcal{P}_\Phi \left( X - \sum_{j=1}^{d} L_j \right) : \Psi \cdot \left( X - \sum_{j=1}^{d} L_j \right)_{r,s \in \Phi} \neq 0
\]

and

\[
\geq \left( X - \sum_{j=1}^{d} L_j \right)_{r,s \in \Phi} , |\Psi| \leq K. \tag{10}
\]

The projection \(S = \mathcal{P}_\Phi (O)\) represents that the matrix \(S\) has the same entries as \(O\) on the index set \(\Psi\), while the other entries are all zeros.

The decomposition is then obtained by iteratively solving these \(d + 1\) subproblems in (7) by following (9) and (10). In this paper, we initialize \(L^i_{\Omega_1}\) and \(S\) as

\[
\begin{align*}
L^i_{\Omega_1} & := Z_{\Omega_1} , i = 1, \ldots , d, \\
Z & := D^{-1} X , D = \text{diag} \left( Y \bar{T}^T \right) ; \\
S & := 0 . \tag{11}
\end{align*}
\]

In each subproblem, only one variable is optimized with the other variables fixed. The convergence of such kind of alternating minimization to a stationary point has been proved. Moreover, a linear convergence rate result can be established by applying the similar technique in GoDec [22].

After obtaining the decomposition by solving (6), each training sample is represented by the sum of \(k\) elements in \(L_1^i\) characterized by the labelsets \(i\) is annotated with and the residual in \(S\). Therefore, the mapping of labelset \(D_i\) in feature subspace \(X\) is defined as the row space \(C_i \in \mathbb{R}^{n \times p}\) of the matrix \(L^i_{\Omega_1}\), which can be obtained via the QR decomposition of \(\left( L^i_{\Omega_1} \right)^T\).

### 4.2 Speed up training via bilateral random projections

The main computation in (9) and (10) is the \(d\) times of SVD in updating \(L^i_{\Omega_1}\). SVD requires \(m^3/3 + m^2n\) flops for an \(m \times n\) matrix, and thus it is impractical when \(L^i_{\Omega_1}\) is of large size. Recently, random projection becomes an effective tool in accelerating the matrix multiplication and decomposition [4]. In this paper, we introduce “bilateral random projections (BRP)” to speed up the SVD in the optimization of \(L^i_{\Omega_1}\).

For clear representation, we use notations independent to the ones we use in other parts of this paper to illustrate BRP. In particular, given \(r\) bilateral random projections (BRP) of an \(m \times n\) dense matrix \(X\) (w.l.o.g. \(m \geq n\)), i.e., \(Y_1 = X A_1\) and \(Y_2 = X^T A_2\), wherein \(A_1 \in \mathbb{R}^{m \times r}\) and \(A_2 \in \mathbb{R}^{n \times r}\) are random matrices, \(L = Y_1 (A_1^T Y_1)^{-1} Y_2^T \)

is a fast rank-\(r\) approximation of \(X\). The computation of \(L\) includes an inverse of an \(r \times r\) matrix and three matrix multiplications. Thus, for a dense \(X\), \(2mnr\) floating-point operations (flops) are required to obtain \(BRP\), \(r^2(2n + r) + mnr\) flops are required to obtain \(L\). The computational cost is much less than that of the SVD based approximation, while its approximation error approaches to that of SVD based method.

We build the random matrices \(A_1\) and \(A_2\) in an adaptive way. Initially, both \(A_1\) and \(A_2\) are set to standard Gaussian matrices whose entries are independent variables following the standard normal distribution. We firstly compute \(Y_1 = X A_1\), update \(A_2 := Y_1^T Y_2\) and calculate the left random projection as \(Y_2 = X^T A_2\) by using the new \(A_2\), and then we update \(A_1 := Y_2^T\) and calculate the right random projection \(Y_1 = X A_1\) by using the new \(A_1\). This adaptive updating of random matrices requires additional flops of \(mnr\).

Algorithm 2 summarizes the training stage of LASE with BRP based acceleration.

### Algorithm 2: LASE Training

**Input:** \(X, \Omega_1, r, i = 1, \ldots , d, K, \epsilon\)

**Output:** \(C_i, i = 1, \ldots , d\)

Initialize \(L_i^i\) and \(S\) according to (11), \(t := 0;\)

while not converge do

\[ t := t + 1; \]

for \(i \leftarrow 1 \) to \(d\) do

\[ N := \left( I - \left( |\Omega| + \frac{|\Omega|^2}{\lambda n M^2} \right)^{-1} \bar{T} \bar{T}^T \right) \left( P^i + \frac{1}{2} Q^i \right); \]

Generate standard Gaussian matrix \(A_1 \in \mathbb{R}^{m \times r}\):

\[ Y_1 := N A_1, A_2 := Y_1; \]

\[ Y_2 := N^T Y_1, Y_1 := N Y_2; \]

\[ L^i_{\Omega_1} := Y_1 (A_1^T Y_1)^{-1} Y_2^T, L^i_{\Omega_1} := 0; \]

end

\[ N := \left| X - \sum_{j=1}^{K} L^j \right|; \]

\[ S := \mathcal{P}_\Phi (N), \Psi \text{ is the index set of the } K \text{ largest entries of } N; \]

end

Do QR decomposition of \(\left( L^i_{\Omega_1} \right)^T\) for each \(i = 1, \ldots, d\) and assign the obtained orthogonal bases to \(C_i\).

### 5. PREDICTION BY GROUP LASSO

In this section, we introduce the prediction stage of LASE by estimating the group sparse representation of a given instance on the obtained subspace ensemble \(C\). Notice that in the training stage, we decompose the training data into the sum of low-rank components \(L^i_{\Omega_1}\), characterized by the labelsets \(D_i\) and a sparse residual \(S\). The mapping of labelset \(D_i\) in the feature space is defined as the row space \(C_i \in \mathbb{R}^{n \times p}\) of the matrix \(L^i_{\Omega_1}\), and the components of the training instances characterized by labelset \(D_i\) lies in the linear subspace \(C_i\).
In the prediction of LASE, we use group lasso \[19\] to estimate the group sparse representation \( \beta \in \mathbb{R}^{\sum r} \) of an instance \( x \in \mathbb{R}^p \) on the subspace ensemble \( C = [C_1; \ldots; C_d] \), wherein the \( d \) groups are defined as the \( d \) index sets for the coefficients on \( C_1, \ldots, C_d \). Since group \( lasso \) selects nonzero coefficients group-wisely, nonzero coefficients will concentrate on the groups corresponding to the labelsets that the instance belongs to.

According to the above analysis, we solve the following group \( lasso \) problem in the prediction stage of LASE

\[
\min_{\beta} \frac{1}{2} \| x - \beta C \|^2_2 + \rho \sum_{i=1}^d \| \beta G_i \|_2,
\]

where the index set \( G_i \) includes all the integers between \( 1 + \sum_{j=1}^{i-1} r^j \) and \( \sum_{j=1}^{i} r^j \) (including them).

To obtain the final prediction of the label vector \( y \in \{0, 1\}^k \) for a test sample \( x \), we use a simple thresholding of the magnitude sum of coefficients in each group to test which groups that the sparse coefficients in \( \beta \) concentrate on

\[
y = \bigcup_{i \in \Omega} D_i, \quad \Omega = \{ i : \| \beta G_i \|_1 \geq \delta \}.
\]

Although \( y \) can also be obtained via selecting the groups with nonzero coefficients when \( \rho \) in (13) is chosen properly, we set the threshold \( \delta \) as a small positive value to guarantee the robustness to \( \rho \).

Algorithm 3 summarizes the prediction stage of LASE.

**Algorithm 3: LASE Prediction**

**Input:** \( x, D, C^i, i = 1, \ldots, d, \rho, \delta \)

**Output:** \( y \)

Solve group \( lasso \) in (13) by using a group \( lasso \) solver; Predict \( y \) via thresholding in (14).

6. EXPERIMENTS

In this section, we evaluate LASE on several benchmark multimedia datasets for multi-label annotation including of texts, images, video, scene, music, genomics and web page data. We compare LASE with BR [12], ML-KNN [20] and MDDM [21] on four evaluation metrics for evaluating the effectiveness, as well as the CPU seconds for evaluating the efficiency. All the experiment are run in MatLab on a server with 3.33 GHz Intel Xeon processors and 32 GB RAM. In the experiments of multi-label prediction, four
metrics, which are Hamming loss, precision, recall, F1 score and accuracy, are used to measure the prediction performance. The detailed definitions of these metrics are given in Section 7.1.1 of [13].

These four metrics have been broadly applied on general binary data. However, their importances are different to each other in evaluating the performance of multi-label prediction, because there are much more 1s than 0s in the label matrix. Precision and recall should be considered together, because high precision always accompanies low recall when most positive samples are falsely predicted as positive. F1-score and accuracy are less sensitive to the imbalance of label matrix. Therefore, a fair evaluation of prediction performance should include integrative consideration of all the four metrics, whose importances can be roughly given by $F_1$, Accuracy $\geq \{Precision, Recall\}$.

### 6.1 Datasets

We evaluate the prediction performance and time cost of LASE on 9 datasets from different domains and of different scales, including Corel5k (image), Mediamill (video), Enron (text), Bibtex (text), Slashdot (text) and 4 sub dataset selected in Yahoo dataset (web page data). The 101 concepts of Mediamill and the sample images of Corel5k are shown in Figure 3. These datasets were obtained from Mulan’s website 1 and MEKA’s website 2. They were collected from different practical multi-label annotation problems. Table 1 shows the number of samples $n$ (training samples+test samples), number of features $p$, number of labels $k$, and the average cardinality of all label vectors “Card” of different datasets.

### 6.2 Performance comparison

We show the prediction performance and time cost in CPU seconds of BR, ML-KNN, MDDM and MSE in Table 2. In BR, we use the MatLab interface of LIBSVM 3 to train the classic linear SVM classifiers for each label. The parameter $C \in \{10^{-3}, 10^{-2}, 0.1, 1, 10, 10^2, 10^3\}$ with the best performance on the training set was used. In ML-KNN, the number of neighbors was 30 for all the datasets.

In MDDM, the regularization parameter for uncorrelated subspace dimensionality reduction was selected as 0.12 and the dimension of the subspace was set as 20% of the dimension of the original data. In LASE, we selected $\lambda = \frac{\gamma}{|\Omega|}$ with $\gamma \in [0.03, 0.3]$, $K \in [10^{-6}np, 10^{-3}np]$, $\lambda \in [0.05, 1.5]$, $\rho \in [0.2, 0.45]$ and $\delta \in [10^{-4}, 10^{-2}]$. For the datasets such as Bibtex, Corel5k, Mediamill and Enron with ambient label space, that is, with large $k$, we find the mappings of labelsets $D$ obtained by LD in LASE. For other datasets, we directly study the mappings of the original labels. For each dataset, we roughly selected 4 groups of parameters in the above ranges and chose the one with the best performance on the training data. Group ‘lasso’ in LASE is solved by SLEP [6] in our experiments.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
ID & Datasets & $n$ (training samples+test samples) & $p$ & $k$ & “Card” \\
\hline
1 & Bibtex & 4880 + 2515 & 1836 & 159 & 2.402 \\
2 & Corel5k & 4500 + 300 & 499 & 374 & 3.522 \\
3 & Mediamill & 30953 + 12014 & 120 & 101 & 4.376 \\
4 & Enron & 1123 + 579 & 1001 & 53 & 3.378 \\
5 & Slashdot & 2338 + 1444 & 1079 & 22 & 1.181 \\
6 & Computers & 2000 + 3000 & 681 & 33 & 1.508 \\
7 & Recreation & 2000 + 3000 & 606 & 22 & 1.423 \\
8 & Science & 2000 + 3000 & 745 & 40 & 1.451 \\
9 & Society & 2000 + 3000 & 636 & 27 & 1.692 \\
\hline
\end{tabular}
\end{table}

The experimental results show that LASE is competitive on both speed and prediction performance, because it explores label correlations and structure without increasing the problem size. In addition, the bilateral random projections further accelerate the computation. In particular, its training time increases much more slowly than other methods, so it is more efficient when applied to large scale datasets such as Bibtex, Mediamill and Slashdot. MDDM is faster than LASE on a few datasets because MDDM invokes ML-knn on the data after dimension reduction, while LASE is directly applicable to the original high dimensional data.

In the comparison of performance via the four metrics, the F1 score and accuracy of LASE outperform those of other methods on most datasets. Moreover, according to Figure 2, LASE has smaller gaps between precision and recall on different tasks than
other methods, and this implies it is robust to the imbalance between positive and negative samples. Note in multi-label prediction, only large values of all four metrics are sufficient to indicate the success of the prediction, while the combination of some large valued metrics and some small valued ones are always caused by the imbalance of the samples. Therefore, LASE provides better prediction performance than other methods on most datasets.

8. ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for the constructive comments. This work is supported by the Australian Research Council discovery project (ARC DP-120103730).

9. REFERENCES


7. CONCLUSION

In this paper, we develop a novel multi-label annotation method “labelsets anchored subspace ensemble (LASE)” for multimedia retrieval, whose training stage maps each concept group (i.e., labelset obtained by label distilling (LD)) to the feature space as a subspace and conversely use the obtained subspace ensemble to explain the instance by the labelsets it is annotated with. The subspace ensemble not only has the capability to decompose the instance into components explained by different labelsets, but also preserves the geometry of the labelsets after the mapping. The prediction in LASE is formulated as the estimation of group sparse representation for given instances on the subspace ensemble.

The main significance of LASE is that it eliminates the rapid growth of model/sample complexity caused by the increase of the number of labels, which is a common problem of existing multi-label annotation approaches. Thus it is competitive in speed. Meanwhile, the correlations between different concepts are fully explored by LD and the manifold regularization, which together bring a significant improvement to the prediction. Furthermore, LASE provides a new viewpoint to the multi-label annotation problem, which overcomes the drawbacks of classification based methods.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
<th>CPU sec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>1.0</td>
<td>0.85</td>
<td>0.9</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>ML-knn</td>
<td>0.6</td>
<td>0.9</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>MDDM</td>
<td>0.3</td>
<td>0.9</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>LASE</td>
<td>0.7</td>
<td>0.9</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>BR</td>
<td>2.0</td>
<td>0.9</td>
<td>1.9</td>
<td>1.9</td>
<td>1.9</td>
</tr>
<tr>
<td>ML-knn</td>
<td>1.0</td>
<td>0.5</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>MDDM</td>
<td>1.0</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>LASE</td>
<td>0.8</td>
<td>0.9</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>BR</td>
<td>5.0</td>
<td>0.5</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>ML-knn</td>
<td>4.0</td>
<td>0.9</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>MDDM</td>
<td>3.0</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>LASE</td>
<td>4.0</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>BR</td>
<td>1.0</td>
<td>0.5</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>ML-knn</td>
<td>0.5</td>
<td>0.9</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>MDDM</td>
<td>0.3</td>
<td>0.9</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>LASE</td>
<td>0.7</td>
<td>0.9</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>BR</td>
<td>1.0</td>
<td>0.5</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>ML-knn</td>
<td>0.5</td>
<td>0.9</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>MDDM</td>
<td>0.3</td>
<td>0.9</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>LASE</td>
<td>0.7</td>
<td>0.9</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Note: The table shows the prediction performances (%) and CPU seconds of BR, ML-knn, MDDM, and LASE on 9 datasets.