Neurocomputing 501 (2022) 212–221

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Label distribution learning through exploring nonnegative components

Tianyue Zhang [a,c], Yingke Mao [b], Furao Shen [a,d,*], Jian Zhao [e,*]

[a] State Key Laboratory for Novel Software Technology, Nanjing University, China
[b] State Grid Shanghai Maintenance Company, 600 Wuning Road, Putuo District, Shanghai, 200063, China
[c] Department of Computer Science and Technology, Nanjing University, China
[d] School of Artificial Intelligence, Nanjing University, China
[e] School of Electronic Science and Engineering, Nanjing University, Nanjing, China

ARTICLE INFO

Article history:
Received 25 September 2021
Revised 18 April 2022
Accepted 6 June 2022
Available online 9 June 2022
Communicated by Zidong Wang

Keywords:
Label distribution learning
Nonnegative components learning

ABSTRACT

Label distribution learning (LDL) is a new machine learning paradigm to solve label ambiguity and has drawn increasing attention in recent years. The importance of all labels needs to be considered under the LDL settings. A series of approaches have been proposed to deal with the LDL problem by considering the correlation of labels or instances. However, none of them focuses on finding interpretable bases to reduce the dimensions of the feature space. Inspired by the semi-nonnegative matrix factorization (semi-NMF) method, we propose a new LDL learning framework to deal with the problem through learning nonnegative components. The key insight is to explore the bases, each of which represents a class, through the label distribution and to transform the input matrix into a coefficient matrix of the space constructed by the bases. Consequently, a maximum entropy model can be adopted to learn the label distribution from the coefficient matrix. Experimental results on real-world datasets comparing our method with several state-of-the-art methods validate the performance of our approach.

1. Introduction

Generally, single-label learning (SLL) tackles the problem of predicting the *best* label to describe an instance. Recently, researchers have observed that data generated in different fields are not only associated with one label, but also related to several relevant labels. In other words, they are essentially multi-labeled. To handle such tasks, the multi-label learning (MLL) paradigm can provide *a subset of* labels to describe an instance. Numerous algorithms and theoretical analysis have emerged and been successfully applied in real-world applications, such as emotion classification for texts [1] and recommendation system [2].

Although the SLL and MLL learning frameworks have been developed for many years, they also have their limits. They are only concerned with the problem of describing an instance with only one label or subset of labels. However, different labels often describe the instance in different degrees, and the degrees of label importance are of interest in some fields, such as emotion analysis [3]. Recently, a novel learning paradigm named label distribution learning (LDL) [4] has been proposed to deal with this issue by learning labels and the degrees of label importance simultaneously.

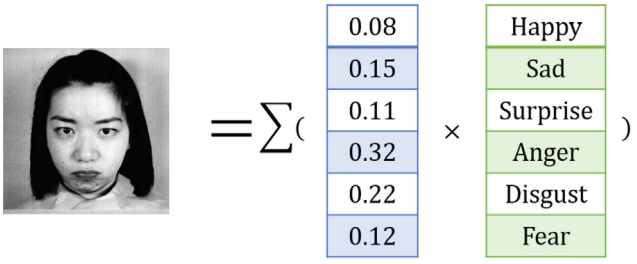
A growing number of researchers have focused on the LDL problem. The representative paper [4] gave an explicit definition of the LDL problem and provided analysis and experiments on the baselines. Besides, a series of typical works were also put forward for LDL. To name a few, IIS-LLD [5] and BFGS-LLD [4] were proposed to learn label distribution for age estimation. Approaches [6] [7] [8] employed similar models by exploring more on the label or instance correlations. Another work named LALOT [9] revealed label correlations based on the optimal transport theory. Duo-LDL [10] is a novel method which employed three-layer MLP to explore relation of different labels.Some typical methods were also adapted for the LDL problem, such as PT-SVM [11] and LDLM [12] that were inspired from SVM methods, AA-kNN [13] and AA-BP [5] that were adapted from kNN and a three-layer neural network. Recently, deep neural networks have been combined with the LDL problem. Several works were proposed such as [14] [15], which especially focused on age estimation applications. Besides, more works have been presented and applied in real-world scenarios, such as emotional analysis [16] [17], facial pose estimation [18] [19] and video parsing [20]. However, most previous works learn a classification model directly from the original data, without reducing the dimensionality of the feature space with the help of dimen-

* Corresponding authors.
E-mail addresses: njucszty@gmail.com (T. Zhang), 35795128@qq.com (Y. Mao), frshen@nju.edu.cn (F. Shen), jianzhao@nju.edu.cn (J. Zhao).

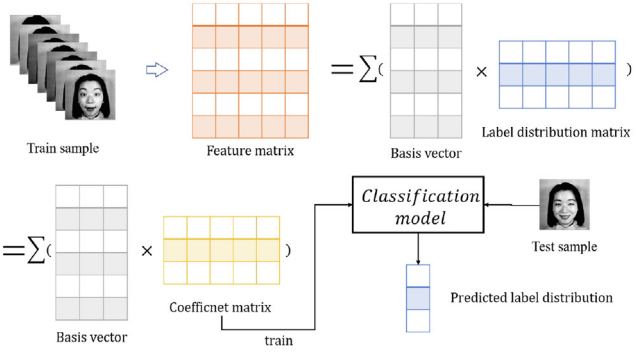
https://doi.org/10.1016/j.neucom.2022.06.017
0925-2312/

sionality reduction methods. Thus, the classification model (such as the maximum entropy model) may learn a better mapping function in the low-dimensional embedded space.

In this paper, we aim to provide a specific method to deal with the LDL problem through learning meaningful components for the feature spaces of the input instances, which is inspired by semi-NMF methods. We assume that every class has at least one *representative* feature, which is represented by a basis. The basis can be fully described by an individual label. Other features from the input space can be regarded as a combination of these bases. Take the emotion analysis as an example to illustrate the problem. An emotional face, which is shown in Fig. 1(a), may be composed of several basic emotions, such as happiness, sadness, surprise, and fear. We aim to find the feature representations of basic emotions, and each mixed emotion can be represented as a combination of the basic features. Therefore, the basis matrix is computed to represent these basic emotions, guided by the label distribution of the training examples. After that, the input features are transformed into a coefficient matrix of the basis space, and the classification model is trained on these coefficients instead of the original features. The process is demonstrated in Fig. 1(b). On the one hand, the dimensionality of the label space is often smaller than that of the feature space on most datasets. Thus, we transform the complex problem of learning in the high-dimensional feature space into a simpler problem that learns the classifier for the coefficient matrix, which can provide better results. On the other hand, nonnegative basis vectors can provide interpretable factors for every class. NMF methods assume that every feature can be reconstructed by adding up the basis with weights, while we relax the nonnegative constraints of coefficient matrix for the reason that subtraction exists in nature. It is also noteworthy to mention that our approach reduces the requirement of prior knowledge because we do not need to set the dimensionality of the embedded basis space. Thus, no more parameters are introduced in the procedure of learning the basis matrix and the coefficient matrix, which is difficult to determine in real-world applications.



(a) This emotion is a mixture of several basic emotions, including happy, sad, surprise, anger, disgust and fear. The degrees of every emotion is normalized to add up to 1.



(b) The process of our method. We aim to reveal the representation of the basic emotion (i.e. basis), and transform the emotion instances into combination of the basis vectors, which is described by a coefficient matrix. Then the classification model will be trained on the matrix.

Fig. 1. An emotion figure example illustrates the label distribution problem and the key idea of our approach.

The main contributions of this paper are summarized in the following three points:

(1) We propose the method for learning the basis and coefficient matrices for the original input features. The method not only reduces the dimensionality of feature space for most datasets, but also explores interpretable bases.
(2) Combining the above method and maximum entropy model, we propose a novel LDL learning framework through learning nonnegative components which is named LDL-ENC. No more parameters are imported into our new method so that no prior knowledge is required for setting parameters.
(3) The optimization procedure and analysis on experiments are provided in this paper. Experimental results on 13 datasets demonstrate that the proposed method is competitive with the state-of-the-art LDL methods. Analysis on the optimization methods and the computational time are also reported.

In the following sections, we start with a brief review of related works. Then, the learning and optimizing algorithm will be introduced. Next, experimental results on real-world datasets and analysis on results are reported. Finally, the conclusion is given.

2. Related Works

Linear dimensionality reduction (DR) is a common technique in the field of data analysis, and nonnegative matrix factorization (NMF) is a widely used DR method. It has been widely employed and developed in recent years, and applied in real-world applications such as image processing, text mining and hyperspectral imaging [21]. NMF constructs a nonnegative feature space and reconstructs samples with linear combination of meaningful bases in this space.

NMF basically tries to decompose a data matrix $\mathbf{X}$ by minimizing $||\mathbf{X}-\mathbf{BU}||_F$, where $\mathbf{B}\geqslant 0, \mathbf{U}\geqslant 0$ (here $\geqslant$means componentwise nonnegative). $\mathbf{X}, \mathbf{B}$ and $\mathbf{U}$ denote the feature matrix, the basis matrix and the coefficient matrix respectively. The objective function is nonconvex when both two variables $\mathbf{B}$ and $\mathbf{U}$ are needed to be optimized, and it is often solved by alternatively optimizing on one of the two matrices. Multiplicative updates (MU), alternating least squares method (ALS), alternating nonnegative least squares (ANLS) and more optimization methods are employed to decompose the feature matrix [21]. Some previous works relax the nonnegative constraints for wider applications, and these NMF methods are named as semi-NMF [22] [23].

Importing label information into the NMF method has also drawn researchers' attention. Several supervised or semi-supervised NMF methods [24] [25] were proposed. However, they solve the SLL or MLL problem and are not suitable for the LDL problem. In this paper, we focus on exploiting the insight of NMF method and combine it to solve the supervised LDL problem.

3. The approach

3.1. Problem Formulation

Let $\mathbf{X} \in \mathbb{R}^{m\times n}$ denote the input feature matrix where n is the number of the m-dimensional instances, $\mathscr{L} = \{l_1, l_2, \ldots, l_c\}$ denote the complete set of c labels, $d_{\boldsymbol{x}_i}^{l_j}$ (briefly denoted as d_i^j) denote the description level of the label $l_j \in \mathscr{L}$ to the ith instance $\boldsymbol{x}_i$, and $\mathbf{D} \in \mathbb{R}^{c\times n}$ denote the label description distribution matrix of all the c labels for the data $\mathbf{X}$. Two constraints are imposed for the description level, i.e., $d_i^j \in [0,1]$ and $\sum_j d_i^j = 1$. Consequently, the training set is denoted as $\mathscr{S} = \{(\boldsymbol{x}_1, \boldsymbol{d}_1), (\boldsymbol{x}_2, \boldsymbol{d}_2), .., (\boldsymbol{x}_n, \boldsymbol{d}_n)\}$. Our goal is predicting the label distribution for unseen instances. In this paper, the predicted distribution matrix from the trained model is denoted as $\mathbf{Y} \in \mathbb{R}^{c\times n}$, in which y_i^j is the predicted description level for instance $\boldsymbol{x}_i$ on label l_j. Several measure metrics will evaluate the difference between $\mathbf{Y}$ and $\mathbf{D}$ in experiments.

3.2. Nonnegative Components Learning

As mentioned earlier, we design LDL-ENC to discover the interpretable basis for every label. Formally, we denote the basis vector as $\boldsymbol{b}_j$ for the label j. If we regard the bases as features, the description level of its labels satisfies such conditions: $d_{\boldsymbol{b}_j}^j = 1$ and $d_{\boldsymbol{b}_j}^{q\neq j} = 0$. To learn the interpretable and natural bases, we require the basis $\boldsymbol{b}$ to be nonnegative, i.e., $\boldsymbol{b} \geqslant 0$.

Firstly, our goal is to learn the basis matrix. Intuitively, the instance $\boldsymbol{x}_i$ can be represented as a linear combination of the bases. In other words, instances can be reconstructed approximately with the bases, and we choose the best basis for each class.

Fortunately, the label distribution provides the description level of labels for instances in the LDL problem, which can simplify the complex procedure of alternatively updating. More precisely, we assume that the description level represents the degree of importance for bases to construct the features. Therefore, we can learn the bases through the feature matrix and the label distribution matrix, i.e., we use label distribution as the initial coefficient matrix. The learning process is described as

$$\boldsymbol{x}_i = \sum_{j=1}^{c} d_i^j \boldsymbol{b}_j. \tag{1}$$

Accordingly, the basis matrix is computed through minimizing the Frobenius norm of the difference between the reconstruction matrix and feature matrix, i.e., by minimizing the reconstruction error as

$$\begin{aligned} T_B(\mathbf{B}) = \quad & \min_{\mathbf{B}} ||\mathbf{X} - \mathbf{B}\mathbf{D}||_F^2 \\ s.t. \quad & \mathbf{B} \geqslant 0 \end{aligned} \tag{2}$$

where $\mathbf{B} \in \mathbb{R}^{m\times c}$ denotes the basis matrix. The problem (2) can be regarded as optimizing the sum of least squares for all input instances in the datasets.

With the bases learned, we can transform the input instances from the original feature space to the basis space. Considering for the basis matrix, the label distribution matrix may not be the optimal solution for minimizing the reconstruction error, so we also need to compute the coefficient matrix $\mathbf{U} \in \mathbb{R}^{c\times n}$ with $\mathbf{X}$ and $\mathbf{B}$ fixed. Besides, subtraction is also natural and interpretable, so we do not require the coefficient matrix to be nonnegative. Thus, we can obtain the representative coefficients $\mathbf{U}$ through minimizing the reconstruction error, i.e.,

$$T_U(\mathbf{U}) = \min_{\mathbf{U}} ||\mathbf{X} - \mathbf{B}\mathbf{U}||_F^2. \tag{3}$$

Through learning the basis matrix and the coefficient matrix, the input features are now represented in the embedded basis space. Comparing $\mathbf{X} \in \mathbb{R}^{m\times n}$ with $\mathbf{U} \in \mathbb{R}^{c\times n}$, the dimensionality is reduced when $c < m$, i.e., the dimensionality of the label space is less than that of the feature space.

In summary, learning nonnegative components helps us to reduce the dimensionality of the feature space, and label distribution provides us with more information, which makes our approach more efficient than unsupervised NMF, i.e., our method only needs one-pass optimization without alternate updating repeatedly.

3.3. Classification Model

With the coefficient matrix, we can now learn the classification model to predict the label distribution. Similar to previous works [4] [7], we assume it follows a maximum entropy model $p(\mathbf{Y}|\mathbf{U};\boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \mathbb{R}^{c\times c}$. Thus, the model for predicting the description level of l_j of the coefficient $\boldsymbol{u}_i$ can be represented as

$$\begin{aligned} p(y_i^j|\boldsymbol{u}_i;\boldsymbol{\theta}) &= \tfrac{1}{Z_i} \exp(\boldsymbol{\theta}_j \boldsymbol{u}_i) \\ Z_i &= \sum_{j=1}^{c} \exp(\boldsymbol{\theta}_j \boldsymbol{u}_i). \end{aligned} \tag{4}$$

To construct the loss function of learning classification model, the Kullback–Leibler (KL) divergence is adopted as the distance measure, which is common for evaluating the distance between two distributions and ranges in $[0,\infty]$. Smaller KL indicates that the two distributions are more similar. Thus, the loss function of model parameter θ is determined as

$$T_\theta(\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{j=1}^{c} d_i^j \ln \frac{d_i^j}{p(y_i^j|\boldsymbol{u}_i;\boldsymbol{\theta})}. \tag{5}$$

Leaving out the fixed value, the loss function can be computed as

$$\begin{aligned} T_\theta(\boldsymbol{\theta}) &= \sum_{i=1}^{n}\sum_{j=1}^{c} d_i^j \ln \frac{\sum_{j=1}^{c} \exp(\boldsymbol{\theta}_j \boldsymbol{u}_i)}{\exp(\boldsymbol{\theta}_j \boldsymbol{u}_i)} \\ &= \sum_{i=1}^{n} \ln \sum_{j=1}^{c} \exp(\boldsymbol{\theta}_j \boldsymbol{u}_i) - \sum_{i=1}^{n}\sum_{j=1}^{c} d_i^j \boldsymbol{\theta}_j \boldsymbol{u}_i. \end{aligned} \tag{6}$$

3.4. Optimization

There are three variables needed to be optimized in our method, i.e., $\mathbf{B}, \mathbf{U}$, and $\boldsymbol{\theta}$. Fortunately, there is no need for multiple turns of alternate updating. We optimize the loss function through updating matrices $\mathbf{B}, \mathbf{U}, \boldsymbol{\theta}$ one by one.

The optimization of $\mathbf{B}$ is a nonnegative least squares problem (NNLS), which is generally regarded as a subproblem of NMF. We denote the jth row of basis matrix $\mathbf{B}$ as $\boldsymbol{b}^j$, where $\boldsymbol{b}^j \in \mathbb{R}^{1\times c}$. The optimal values of $\boldsymbol{b}^j$ can be found by solving the following NNLS problem

$$\min_{\boldsymbol{b}^j \geqslant 0} ||\mathbf{D}^T \boldsymbol{b}^{jT} - \boldsymbol{x}^{jT}||_2^2 \tag{7}$$

where $\boldsymbol{x}^j \in \mathbb{R}^{1\times n}$ is the j feature of all input instances.

$$\begin{cases} \mathbf{D}(\mathbf{D}^T\boldsymbol{b}^{jT} - \boldsymbol{x}^{jT}) - \boldsymbol{\lambda} = 0 \\ \boldsymbol{b}^{jT} \geqslant 0 \\ \boldsymbol{\lambda} \geqslant 0 \\ \lambda_i b_i^{jT} = 0, i = 1, \ldots, c \end{cases} \tag{8}$$

$\boldsymbol{b}^j$ **B**The active set algorithm is employed to find the solution quickly, and the details can be found in this paper [26]. For **U**, the problem can be easier to solve with the closed form solution for least squares problem, i.e,

$$\mathbf{U} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{X} \tag{9}$$

For unconstrained optimization of $\boldsymbol{\theta}$, we adopt the effective limited-memory quasi-Newton method (L-BFGS) [27] to optimize our method quickly without computing the inverse Hessian matrix. It is critical to calculate the first-order gradient of objective functions for computing L-BFGS method. The gradient of the objective function $T_\theta(\boldsymbol{\theta})$ is computed as

$$\frac{\partial T_\theta(\boldsymbol{\theta})}{\partial \theta_j^k} = \sum_{i=1}^{n} \frac{\exp(\boldsymbol{\theta}_j \boldsymbol{u}_i) u_i^k}{\sum_{j=1}^{c} \exp(\boldsymbol{\theta}_j \boldsymbol{u}_i)} - \sum_{i=1}^{n} d_i^j u_i^k. \tag{10}$$

Through optimization, our approach can converge and learn the label distribution. We summarize the detailed training and testing procedure of our approach in Algorithm 1 and Algorithm 2.

Algorithm 1: Our approach: training phase

Input: Training set $\mathscr{S}$.
Output: the model parameters **B** and θ.
1: Optimize the basis matrix **B** using active set algorithm;
2: Compute the optimal solution for **U** with Eq. (9);
3: Initialize the maximum entropy model parameter $\boldsymbol{\theta}$ and optimize it using L-BFGS. The $\nabla T_\theta(\boldsymbol{\theta}^{(l)})$ is computed according to Eq. (10);
4: **return** parameters **B** and $\boldsymbol{\theta}$.

Algorithm 2: Our approach: testing phase

Input: test set $\mathscr{X}_{test} = \{\boldsymbol{x}_{test1}, \boldsymbol{x}_{test2}, .., \boldsymbol{x}_{testn}\}$, denoted as matirx $\mathbf{X}_{test}$.
Output: the predicted label distribution $p(\mathbf{Y}|\mathbf{X}_{test}, \boldsymbol{\theta})$.
1: Compute the optimal solution for $\mathbf{U}_{test}$ with Eq. (9);
2: Predict $p(\mathbf{Y}|\mathbf{X}_{test}, \boldsymbol{\theta})$ using Eq. (4).
3: **return** the predicted label distribution $p(\mathbf{Y}|\mathbf{X}_{test}, \boldsymbol{\theta})$.

Besides, we employ SDPT3 [28] as another optimizing method for solving matrix **B** and **U**, which is realized in the CVX package provided in [29] [30], to compare the results on different optimizing methods. The results are reported in the following experiment section.

4. Experiments

4.1. Datasets

We choose 13 real-world datasets from different fields. The 1st to 10th datasets were collected from the biological experiments [31] on the yeast genes over a period of time yielding different gene expression levels on a series of time points. sJAFFE is a facial expression dataset collected from 10 Japanese female models [32], including 6 basic facial expressions as labels: happy, sad, surprise, anger, disgust, fear, and each image was rated on 6 emotion adjectives by 60 Japanese subjects. The rates are normalized and 243-dimensional features are extracted by Local Binary Patterns (LBP) [33]. Human Gene is a large-scale real-world dataset on the relationship between human genes and diseases. Each gene is represented as a 36-dimensional descriptor vector and corresponds to 68 diseases. The Movie dataset includes ratings for 7755 movies that come from Netflix, ranging from 1 to 5. The distribution of ratings is calculated from the percentage of each level, and the 1869-dimensional features are extracted from the metadata of the movies. The details of the datasets are summarized in Table 1.

4.2. Evaluation Measures

We choose six different distance metrics to evaluate the performance of our method, including *Cosine*, *Chebyshev*, *KL*, *Euclidean*, *Canberra* and *Intersection*. The cosine and intersection metrics evaluate the similarity between two distributions, indicating that the larger the performance the better. Other four metrics, Chebyshev, Euclidean, KL divergence and Canberra, evaluate the distance between two distributions, indicating that the smaller the performance the better. The formulas of these measures are presented as

$$Euclidean(\boldsymbol{D}_i, \boldsymbol{y}_i) = \sqrt{\sum_{j=1}^{c} (d_i^j - y_i^j)^2} \tag{11}$$

$$Cosine(\boldsymbol{D}_i, \boldsymbol{y}_i) = \frac{\sum_{j=1}^{c} d_i^j y_i^j}{||\boldsymbol{d}_i||_2 ||\boldsymbol{y}_i||_2} \tag{12}$$

$$KL(\boldsymbol{D}_i, \boldsymbol{y}_i) = \sum_{j=1}^{c} d_i^j \ln \frac{d_i^j}{y_i^j} \tag{13}$$

$$Chebyshev(\boldsymbol{D}_i, \boldsymbol{y}_i) = \max_j |d_i^j - y_i^j| \tag{14}$$

$$Canberra(\boldsymbol{D}_i, \boldsymbol{y}_i) = \sum_{j=1}^{c} \frac{|d_i^j - y_i^j|}{d_i^j + y_i^j} \tag{15}$$

$$Intersection(\boldsymbol{D}_i, \boldsymbol{y}_i) = \sum_{j=1}^{c} \min(d_i^j, y_i^j). \tag{16}$$

4.3. Baselines and Settings

Six state-of-art LDL algorithms are compared with our approach, including AA-kNN [13], AA-BP [5], IIS-LLD [5], BFGS-LLD [4], LALOT [9] and Duo-LDL [10]. AA-kNN and AA-BP are two adapted algorithms, which employ k-NN and three-layer neural network, and soft their outputs to adapt them to the LDL problem. IIS-LLD and BFGS-LLD are two specialized methods, while IIS-LLD optimizes the model with the Gauss–Newton method and the latter uses the L-BFGS method. They construct the same loss function

Table 1
Statistics of the 13 datasets used in the experiments.

Index	Dataset	#features	#label	#instances
1	Yeast-alpha	24	18	2465
2	Yeast-cdc	24	15	2465
3	Yeast-cold	24	4	2465
4	Yeast-diau	24	14	2465
5	Yeast-dtt	24	4	2465
6	Yeast-elu	24	14	2465
7	Yeast-heat	24	6	2465
8	Yeast-spo	24	6	2465
9	Yeast-spoem	24	2	2465
10	Yeast-spo5	24	3	2465
11	sJAFFE	243	6	213
12	Human Gene	36	68	30542
13	Movie	1869	5	7755

Table 2
Experimental results are measured by Chebyshev and Cosine distances of different LDL algorithms on LDL datasets, and the last rows in subtables indicate the average ranking of each method on all datasets.

(a) Results (mean ± std) are measured by Chebyshev distance ↓, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	ENC-SDPT3
Alpha	.0147 ± .0008(5)	.0401 ± .0022(8)	.0156 ± .0004(6)	.0138 ± .0002(4)	.0160 ± .0010(7)	.0135 ± .0003(3)	**.0134 ± .0004(1)**	**.0134 ± .0004(1)**
Cdc	.0173 ± .0005(6)	.0411 ± .0022(8)	.0184 ± .0005(7)	.0170 ± .0007(5)	.0165 ± .0005(3)	.0165 ± .0007(3)	**.0162 ± .0005(1)**	**.0162 ± .0005(1)**
Cold	.0542 ± .0017(5)	.0598 ± .0031(8)	.0545 ± .0017(7)	.0543 ± .0028(6)	.0512 ± .0015(3)	.0513 ± .0016(4)	.0510 ± .0018(2)	**.0510 ± .0017(1)**
Diau	.0385 ± .0012(5)	.0531 ± .0053(8)	.0397 ± .0011(6)	.0418 ± .0010(7)	**.0370 ± .0012(1)**	.0374 ± .0011(4)	**.0370 ± .0012(1)**	**.0370 ± .0012(1)**
Dtt	.0385 ± .0013(6)	.0470 ± .0042(8)	.0406 ± .0014(7)	.0374 ± .0014(5)	.0361 ± .0012(3)	.0360 ± .0013(4)	**.0359 ± .0012(1)**	**.0359 ± .0012(1)**
Elu	.0173 ± .0004(6)	.0409 ± .0023(8)	.0186 ± .0004(7)	.0170 ± .0004(5)	.0164 ± .0005(3)	.0165 ± .0005(3)	**.0163 ± .0004(1)**	**.0163 ± .0004(1)**
Heat	.0441 ± .0012(6)	.0534 ± .0035(8)	.0495 ± .0013(7)	.0435 ± .0011(5)	.0422 ± .0013(3)	.0425 ± .0013(4)	**.0422 ± .0012(1)**	**.0422 ± .0012(1)**
Spo	.0627 ± .0023(7)	.0684 ± .0031(8)	.0605 ± .0018(5)	.0606 ± .0020(6)	.0585 ± .0020(3)	.0586 ± .0021(4)	**.0583 ± .0018(1)**	**.0583 ± .0018(1)**
Spoem	.0904 ± .0047(7)	.0892 ± .0049(6)	.0905 ± .0036(8)	.0880 ± .0055(5)	.0871 ± .0037(2)	**.0870 ± .0037(1)**	.0873 ± .0037(3)	.0874 ± .0037(4)
Spo5	.0948 ± .0036(6)	.0949 ± .0036(7)	.0931 ± .0037(4)	**.0908 ± .0037(1)**	.0913 ± .0033(4)	.0914 ± .0040(5)	.0912 ± .0038(2)	.0912 ± .0038(2)
sJAFFE	.1141 ± .0108(3)	.1272 ± .0126(7)	.1194 ± .0130(5)	.1191 ± .0110(6)	.1291 ± .0120(8)	.1142 ± .0132(4)	**.0956 ± .0103(1)**	.0959 ± .0103(2)
Human Gene	.0648 ± .0018(8)	.0624 ± .0019(7)	.0534 ± .0016(4)	.0534 ± .0007(2)	.0534 ± .0007(2)	.0534 ± .0018(5)	.0534 ± .0018(5)	**.0533 ± .0018(1)**
Movie	.1542 ± .0048(7)	.1572 ± .0024(8)	.1508 ± .0016(6)	.1382 ± .0007(5)	.1240 ± .0032(3)	.1355 ± .0018(4)	.1199 ± .0256(2)	**.1197 ± .0024(1)**
Avg.Rank	5.92	7.62	6.00	4.77	3.46	3.69	1.69	1.38

(b) Results (mean ± std) are measured by Cosine distance ↑, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	ENC-SDPT3
Alpha	.9938 ± .0003(6)	.9391 ± .0059(8)	.9927 ± .0003(7)	.9939 ± .0002(5)	**.9946 ± .0003(1)**	.9943 ± .0003(4)	**.9946 ± .0003(1)**	**.9946 ± .0003(1)**
Cdc	.9924 ± .0004(5)	.9508 ± .0047(8)	.9915 ± .0004(7)	.9924 ± .0004(5)	**.9933 ± .0003(1)**	.9928 ± .0004(4)	**.9933 ± .0003(1)**	**.9933 ± .0003(1)**
Cold	.9872 ± .0008(5)	.9844 ± .0016(8)	.9871 ± .0009(6)	.9870 ± .0015(7)	.9886 ± .0008(2)	.9880 ± .0007(4)	.9886 ± .0008(2)	**.9886 ± .0007(1)**
Diau	.9866 ± .0008(5)	.9742 ± .0053(8)	.9861 ± .0007(7)	.9850 ± .0005(6)	**.9879 ± .0007(1)**	.9869 ± .0007(4)	**.9879 ± .0007(1)**	**.9879 ± .0007(1)**
Dtt	.9933 ± .0005(6)	.9898 ± .0021(8)	.9926 ± .0005(7)	.9934 ± .0005(5)	**.9941 ± .0004(1)**	.9940 ± .0005(4)	**.9941 ± .0004(1)**	.9941 ± .0005(3)
Elu	.9931 ± .0002(6)	.9557 ± .0042(8)	.9922 ± .0003(7)	.9934 ± .0001(5)	.9940 ± .0002(2)	.9940 ± .0003(3)	**.9941 ± .0002(1)**	.9940 ± .0003(3)
Heat	.9867 ± .0006(6)	.9782 ± .0030(8)	.9857 ± .0007(7)	.9871 ± .0005(5)	**.9880 ± .0006(1)**	.9878 ± .0006(4)	**.9880 ± .0006(1)**	**.9880 ± .0006(1)**
Spo	.9730 ± .0017(7)	.9679 ± .0029(8)	.9753 ± .0013(5)	.9745 ± .0014(6)	**.9770 ± .0012(1)**	.9768 ± .0013(4)	**.9770 ± .0012(1)**	**.9770 ± .0012(1)**
Spoem	.9764 ± .0023(8)	.9778 ± .0034(6)	.9774 ± .0015(7)	.9778 ± .0023(5)	**.9790 ± .0015(1)**	**.9790 ± .0015(1)**	.9788 ± .0016(3)	.9788 ± .0016(3)
Spo5	.9713 ± .0022(8)	.9723 ± .0019(7)	.9731 ± .0019(6)	**.9741 ± .0007(1)**	.9741 ± .0018(4)	.9741 ± .0016(2)	.9741 ± .0018(4)	.9741 ± .0016(2)
sJAFFE	.9337 ± .0182(3)	.9145 ± .0140(8)	.9314 ± .0104(5)	.9316 ± .0083(4)	.9100 ± .0100(7)	.9301 ± .0121(6)	**.9531 ± .0086(1)**	.9530 ± .0090(2)
Human Gene	.7687 ± .0046(7)	.6906 ± .0087(8)	.8334 ± .0040(5)	.8333 ± .0018(6)	**.8345 ± .0020(1)**	.8342 ± .0039(3)	.8342 ± .0039(3)	.8345 ± .0039(2)
Movie	.8802 ± .0026(8)	.8948 ± .0012(7)	.9067 ± .0023(6)	.9147 ± .0028(5)	.9264 ± .0032(3)	.9231 ± .0028(4)	.9298 ± .0027(2)	**.9299 ± .0032(1)**
Avg.Rank	6.15	7.77	6.30	5.00	2.00	3.62	1.69	1.69

Table 3
Experimental results are measured by Euclidean distance and KL divergence of different LDL algorithms on LDL datasets, and the last rows in subtables indicate the average ranking of each method on all datasets.

(a) Results (mean ± std) are measured by Euclidean distance ↓, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	LDL-ENC-BFGS	LDL-ENC-SDPT3
Alpha	.0249 ± .0005(6)	.0798 ± .0046(8)	.0273 ± .0005(7)	.0245 ± .0004(5)	**.0231 ± .0006(1)**	.0236 ± .0003(4)	**.0231 ± .0006(1)**	**.0231 ± .0006(1)**
Cdc	.0301 ± .0007(6)	.0770 ± .0043(8)	.0320 ± .0007(7)	.0288 ± .0008(5)	**.0279 ± .0007(1)**	.0284 ± .0007(4)	**.0279 ± .0007(1)**	**.0279 ± .0007(1)**
Cold	.0724 ± .0023(6)	.0798 ± .0040(8)	.0728 ± .0024(7)	.0717 ± .0038(5)	.0681 ± .0025(3)	.0691 ± .0024(4)	**.0681 ± .0022(1)**	**.0681 ± .0022(1)**
Diau	.0567 ± .0016(5)	.0793 ± .0078(8)	.0591 ± .0014(6)	.0593 ± .0013(7)	**.0543 ± .0015(1)**	.0546 ± .0016(4)	**.0543 ± .0015(1)**	**.0543 ± .0015(1)**
Dtt	.0512 ± .0016(6)	.0627 ± .0056(8)	.0541 ± .0018(7)	.0511 ± .0017(5)	**.0480 ± .0015(3)**	.0481 ± .0017(4)	**.0479 ± .0017(1)**	**.0479 ± .0017(1)**
Elu	.0298 ± .0005(6)	.0750 ± .0041(8)	.0321 ± .0006(7)	.0293 ± .0004(5)	**.0278 ± .0006(4)**	**.0277 ± .0006(1)**	**.0277 ± .0006(1)**	**.0277 ± .0006(1)**
Heat	.0622 ± .0016(6)	.0792 ± .0051(8)	.0651 ± .0017(7)	.0615 ± .0013(5)	.0593 ± .0016(3)	.0594 ± .0016(4)	**.0592 ± .0015(1)**	**.0592 ± .0015(1)**
Spo	.0880 ± .0027(7)	.0975 ± .0043(8)	.0854 ± .0024(6)	.0851 ± .0029(5)	**.0816 ± .0021(1)**	.0822 ± .0026(4)	.0819 ± .0024(3)	.0819 ± .0022(2)
Spoem	.1279 ± .0066(6)	.1432 ± .0083(8)	.1280 ± .0050(7)	.1244 ± .0078(5)	.1233 ± .0048(2)	**.1231 ± .0050(1)**	.1235 ± .0053(3)	.1235 ± .0053(3)
Spo5	.1216 ± .0046(7)	.1216 ± .0047(8)	.1192 ± .0047(6)	**.1165 ± .0049(1)**	**.1165 ± .0049(1)**	.1170 ± .0040(5)	.1167 ± .0048(4)	.1167 ± .0041(3)
sJAFFE	.1564 ± .0100(7)	.1713 ± .0151(8)	.1531 ± .0131(3)	.1536 ± .0099(4)	.1542 ± .0162(5)	.1549 ± .0146(6)	**.1232 ± .0113(1)**	.1234 ± .0116(2)
Human Gene	.1058 ± .0020(7)	.1272 ± .0028(8)	.0868 ± .0018(6)	.0867 ± .0009(5)	.0864 ± .0015(3)	.0865 ± .0020(4)	.0863 ± .0019(2)	**.0863 ± .0019(1)**
Movie	.2564 ± .0321(8)	.2221 ± .0211(7)	.2004 ± .0145(6)	.1876 ± .0110(5)	.1789 ± .0044(3)	.1819 ± .0033(4)	.1738 ± .0033(2)	**.1736 ± .0023(1)**
Avg.Rank	6.38	7.92	6.31	4.77	2.38	3.79	1.69	1.46

(b) Results (mean ± std) are measured by KL divergence ↓, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	LDL-ENC-BFGS	LDL-ENC-SDPT3
Alpha	.0064 ± .0004(6)	.0771 ± .0084(8)	.0075 ± .0003(7)	.0062 ± .0002(4)	**.0056 ± .0004(1)**	.0063 ± .0003(5)	**.0056 ± .0004(1)**	**.0056 ± .0004(1)**
Cdc	.0083 ± .0005(6)	.0608 ± .0065(8)	.0092 ± .0005(7)	.0078 ± .0004(5)	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**
Cold	.0142 ± .0018(6)	.0174 ± .0026(8)	.0144 ± .0020(7)	.0135 ± .0012(5)	**.0129 ± .0020(1)**	.0130 ± .0014(4)	**.0129 ± .0022(1)**	**.0129 ± .0018(1)**
Diau	.0151 ± .0012(5)	.0299 ± .0069(8)	.0159 ± .0011(6)	.0162 ± .0005(7)	**.0138 ± .0010(1)**	.0139 ± .0012(2)	.0140 ± .0011(3)	.0140 ± .0011(3)
Dtt	.0076 ± .0016(6)	.0114 ± .0027(8)	.0084 ± .0016(7)	**.0067 ± .0004(1)**	.0068 ± .0013(2)	.0071 ± .0015(5)	.0069 ± .0015(3)	.0069 ± .0015(3)
Elu	.0073 ± .0004(6)	.0540 ± .0058(8)	.0083 ± .0004(7)	.0070 ± .0003(5)	**.0065 ± .0004(3)**	.0066 ± .0005(4)	**.0064 ± .0004(1)**	**.0064 ± .0004(1)**
Heat	.0145 ± .0011(6)	.0244 ± .0038(8)	.0156 ± .0012(7)	.0137 ± .0010(5)	.0133 ± .0013(3)	.0135 ± .0012(4)	.0133 ± .0011(2)	**.0133 ± .0010(1)**
Spo	.0303 ± .0021(7)	.0368 ± .0037(8)	.0281 ± .0019(6)	.0272 ± .0021(5)	**.0258 ± .0017(1)**	.0265 ± .0018(4)	.0263 ± .0017(2)	.0263 ± .0018(3)
Spoem	.0291 ± .0037(8)	.0283 ± .0034(6)	.0291 ± .0035(7)	.0256 ± .0032(2)	**.0255 ± .0030(1)**	.0270 ± .0035(3)	.0273 ± .0037(4)	.0273 ± .0038(5)
Spo5	.0343 ± .0031(8)	.0339 ± .0032(7)	.0330 ± .0032(6)	.0295 ± .0024(2)	**.0293 ± .0022(1)**	.0324 ± .0031(5)	.0322 ± .0034(3)	.0322 ± .0034(3)
sJAFFE	.0712 ± .0231(4)	.0960 ± .0183(7)	.0700 ± .0089(3)	.0724 ± .0084(5)	.1061 ± .0112(8)	.0740 ± .0135(6)	**.0500 ± .0090(1)**	**.0500 ± .0090(1)**
Human Gene	.3010 ± .0084(7)	.4691 ± .0169(8)	.2264 ± .0072(3)	.2265 ± .0056(5)	.2358 ± .0110(6)	.2264 ± .0070(2)	.2264 ± .0072(3)	**.2262 ± .0072(1)**
Movie	.2008 ± .0102(7)	.1792 ± .0246(6)	.1368 ± .0121(5)	.4572 ± .0331(8)	**.1131 ± .0625(1)**	.1292 ± .0056(4)	.1210 ± .0049(3)	.1209 ± .0049(2)
Avg.Rank	6.31	7.54	6.00	4.54	2.31	3.77	2.15	2.00

Table 4
Experimental results are measured by Canberra and Intersection distances of different LDL algorithms on LDL datasets, and the last rows in subtables indicate the average ranking of each method on all datasets.

(a) Results (mean ± std) are measured by Canberra distance ↓, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	ENC-SDPT3
Alpha	.7582 ± .0289(6)	2.3521 ± .1282(8)	.7625 ± .0351(7)	.7343 ± .0182(5)	.6813 ± .0186(3)	.6845 ± .0170(4)	**.6812 ± .0174(1)**	.6813 ± .0171(2)
Cdc	.7172 ± .0215(7)	1.7152 ± .1055(8)	.7086 ± .0215(6)	.6871 ± .0183(5)	.6462 ± .0180(2)	.6487 ± .0161(4)	**.6461 ± .0173(1)**	.6466 ± .0157(3)
Cold	.2604 ± .0102(6)	.2681 ± .0143(8)	.2487 ± .0091(7)	.2579 ± .0098(5)	.2408 ± .0092(4)	.2402 ± .0090(3)	.2398 ± .0089(2)	**.2398 ± .0080(1)**
Diau	.4551 ± .0112(6)	.5675 ± .0310(8)	.4487 ± .0170(5)	.4851 ± .0103(7)	**.4331 ± .0100(4)**	.4312 ± .0103(2)	.4312 ± .0140(3)	**.4311 ± .0129(1)**
Dtt	.1821 ± .0071(7)	.2043 ± .0118(8)	.1812 ± .0051(6)	.1772 ± .0071(5)	.1689 ± .0060(3)	.1690 ± .0062(4)	.1687 ± .0065(2)	**.1686 ± .0062(1)**
Elu	.6442 ± .0143(7)	1.4885 ± .0672(8)	.6387 ± .0193(6)	.6253 ± .0152(5)	.5853 ± .0115(4)	.5831 ± .0142(3)	**.5823 ± .0128(1)**	.5825 ± .0130(2)
Heat	.3918 ± .0112(7)	.4589 ± .0286(8)	.3772 ± .0068(5)	.3792 ± .0011(6)	.3646 ± .0100(4)	.3642 ± .0072(2)	.3642 ± .0090(3)	**.3640 ± .0098(1)**
Spo	.5597 ± .0218(7)	.5992 ± .0417(8)	.5231 ± .0312(5)	.5258 ± .0216(6)	.5137 ± .0135(4)	.5133 ± .0145(3)	.5127 ± .0155(2)	**.5126 ± .0144(1)**
Spoem	.1914 ± .0089(8)	.1842 ± .0108(7)	.1840 ± .0099(6)	.1814 ± .0090(5)	.1812 ± .0072(4)	**.1799 ± .0082(1)**	.1808 ± .0092(3)	.1808 ± .0082(2)
Spo5	.2969 ± .0146(8)	.2912 ± .0170(7)	.2871 ± .0191(6)	.2831 ± .0121(5)	**.2821 ± .0100(1)**	.2829 ± .0101(4)	.2823 ± .0115(2)	.2824 ± .0104(3)
sJAFFE	.8431 ± .1131(5)	1.0462 ± .1250(7)	.8751 ± .0842(6)	1.0682 ± .0983(8)	.8142 ± .0700(3)	.8202 ± .0675(4)	**.7108 ± .0553(1)**	.7115 ± .0612(2)
Human Gene	16.2832 ± .8072(7)	22.7847 ± 1.8523(8)	14.5412 ± .6534(6)	14.4873 ± .4323(5)	**14.4423 ± .2176(1)**	14.4532 ± .2207(2)	14.4543 ± .2282(3)	14.4543 ± .2282(3)
Movie	1.2758 ± .0457(7)	1.2693 ± .0872(6)	1.1367 ± .0542(5)	2.2317 ± .1011(8)	1.0772 ± .0201(4)	1.0617 ± .0173(3)	1.0345 ± .0195(2)	**1.0337 ± .0175(1)**
Avg.Rank	6.77	7.62	5.85	5.77	3.15	3.00	2.00	1.77

(b) Results (mean ± std) are measured by Intersection distance ↑, followed by the ranking on this dataset and best in bold.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	ENC-SDPT3
Alpha	.9581 ± .0021(6)	.8772 ± .0081(8)	.9578 ± .0021(7)	.9592 ± .0024(5)	**.9624 ± .0009(1)**	.9621 ± .0009(4)	**.9624 ± .0009(1)**	**.9624 ± .0009(1)**
Cdc	.9528 ± .0028(7)	.8912 ± .0051(8)	.9532 ± .0021(6)	.9541 ± .0010(5)	**.9575 ± .0010(1)**	.9574 ± .0010(4)	**.9575 ± .0010(1)**	**.9575 ± .0010(1)**
Cold	.9362 ± .0024(7)	.9340 ± .0032(8)	.9376 ± .0022(5)	.9363 ± .0020(6)	**.9409 ± .0019(1)**	.9408 ± .0019(4)	.9409 ± .0021(3)	**.9409 ± .0019(1)**
Diau	.9371 ± .0021(6)	.9224 ± .0039(8)	.9381 ± .0020(5)	.9328 ± .0022(7)	.9402 ± .0017(3)	**.9403 ± .0017(1)**	.9402 ± .0017(3)	**.9403 ± .0017(1)**
Dtt	.9549 ± .0021(7)	.9502 ± .0021(8)	.9552 ± .0016(6)	.9563 ± .0015(5)	.9582 ± .0014(4)	.9583 ± .0015(3)	**.9584 ± .0014(1)**	**.9584 ± .0014(1)**
Elu	.9546 ± .0011(7)	.8992 ± .0054(8)	.9547 ± .0011(6)	.9564 ± .0022(5)	**.9591 ± .0010(1)**	.9589 ± .0009(2)	.9589 ± .0015(4)	.9589 ± .0009(2)
Heat	.9362 ± .0023(7)	.9251 ± .0054(8)	.9384 ± .0011(6)	.9871 ± .0005(5)	**.9406 ± .0014(1)**	.9402 ± .0016(4)	.9402 ± .0014(3)	.9403 ± .0016(2)
Spo	.9082 ± .0043(7)	.9022 ± .0069(8)	.9143 ± .0052(5)	.9134 ± .0031(6)	**.9156 ± .0023(1)**	.9155 ± .0023(4)	**.9156 ± .0023(1)**	**.9156 ± .0023(1)**
Spoem	.9072 ± .0043(8)	.9108 ± .0056(7)	.9109 ± .0054(6)	.9126 ± .0044(5)	.9128 ± .0058(2)	**.9131 ± .0038(1)**	.9127 ± .0042(3)	.9126 ± .0037(4)
Spo5	.9044 ± .0051(8)	.9062 ± .0054(7)	.9072 ± .0034(6)	.9088 ± .0043(3)	**.9088 ± .0033(1)**	.9086 ± .0031(5)	.9088 ± .0034(4)	**.9088 ± .0033(1)**
sJAFFE	.8552 ± .0215(4)	.8243 ± .0216(7)	.8513 ± .0147(5)	.8058 ± .0221(8)	.8310 ± .0123(6)	.8606 ± .0121(3)	**.8797 ± .0101(1)**	.8797 ± .0108(2)
Human Gene	.7433 ± .0128(7)	.6712 ± .0221(8)	.7828 ± .0098(6)	.7841 ± .0018(5)	**.7852 ± .0042(1)**	.7846 ± .0034(3)	.7846 ± .0028(2)	.7842 ± .0034(4)
Movie	.7801 ± .0056(7)	.7882 ± .0112(6)	.8004 ± .0100(5)	.6496 ± .0311(8)	.8221 ± .0040(3)	.8192 ± .0054(4)	.8282 ± .0034(2)	**.8284 ± .0032(1)**
Avg.Rank	6.77	7.62	5.69	5.62	2.00	3.23	2.23	1.69

of classification model as our method, yet we transform the input features to an easier-to-learn subspace before optimizing it. LALOT replaces the KL divergence with optimal transport distance to capture the geometry of feature space, and introduces kernel biased regularization into the loss function to explore label correlations. The method requires an alternative optimizing procedure, which may lead to more calculation time. The novel Duo-LDL method exploits a three-layer MLP with $c(c-1)$ output neurons to capture the one-to-one relationship of all the degrees of membership of c classes and to explore the label inter-dependencies, which achieves better results than AA-BP.

We use 10 time 10-fold cross-validation to evaluate these methods and record the mean and standard deviation of their results. Moreover, rankings of these methods on each dataset are also noted and the average rankings are computed according to the performance of methods on all datasets. Our methods have no parameters to set. Parameters for LALOT are set as follows: $\lambda = 0.2$ for most datasets, $\lambda = 2$ for Yeast-spoem, and $\lambda = 0.1$ for s-JAFFE; $C = 200$ for most datasets, $C = 20$ for Yeast-alpha, Yeast-cdc, and Yeast-elu; $\eta = 1e-4$ for all datasets. The number of neighbors k in AA-kNN is set to 5. The number of hidden-layer neurons for AA-BP is set to 60. Parameters for Duo-LDL are the same as the original paper [10].

The results are reported in Table 2, Table 3 and Table 4. Our method using LBFGS is denoted as LDL-ENC-BFGS, and SDPT3 is denoted as LDL-ENC-SDPT3. Analysis of the two optimization methods is discussed in the Section 4.7.

4.4. Results

The experimental results of comparing different methods are presented in Table 2, Table 3 and Table 4. They are presented in the form of $mean \pm std\,(ranking\,on\,the\,dataset)$, and the best results are shown in bold. As can be seen from the average rankings, ENC-BFGS and ENC-SDPT3 outperform all the baselines on most datasets and achieve the best average rankings, which validates that our method is a good choice for solving the LDL problem. The reason why the method behaves poorly on some of the datasets could be that dimensionality reduction omits too much information of the feature space and the single basis for every label may be insufficient for representing the samples. In addition, AA-BP performs poorly on all datasets, AA-kNN and IIS-LLD perform likely. The newly proposed Duo-LDL is also a competitive method, which indicates that the new loss function of Duo-LDL indeed improves the performance compared with AA-BP.

4.5. Validation on performance

In order to verify the effectiveness of learning nonnegative components, we should compare the results of BFGS-LLD method and our method, because the BFGS-LLD method only employs the same maximum entropy model to learn label distribution. Thus, the results of BFGS-LLD can be regarded as ablation experiments of LDL-ENC.

From Table 2, Table 3 and Table 4, we can see that learning nonnegative components really contributes to the overall performance of our method. On most datasets, the performance of our method for all measurements is better than BFGS-LLD. For Yeast-spoem, learning nonnegative components has a negative impact on the classification model as we previously analyzed. For most datasets, the dimensionality of their label sets is smaller than that of feature spaces. Under the circumstances, learning nonnegative components can be seen as dimensionality reduction, and it has a positive effect on the learning of classification models. However, for the Human Gene dataset, the performance is slightly better even when this dataset has more labels than features, which indicates that learning nonnegative components still improves the performance. In this situation, our LDL-ENC method can be treated as a dictionary learning method.

To sum up, learning nonnegative components truly improves the performance of label distribution learning on most datasets compared with only employing classification models. Compared with state-of-the-art methods, our method is a competitive algorithm for solving the LDL problem.

4.6. Computation time

We compare the computation time of our methods with all the comparing methods, and the results are shown in the Table 5. ENC-BFGS takes more time than BFGS-LLD method on 3 datasets, which indicates that the learning of components on these datasets costs more time. On other datasets, the ENC-BFGS spends less time with more learning procedures, which indicates that learning a better embedded subspace could reduce the computation time of following classification training.

Compared with other baselines, ENC-BFGS is competitive in computational time cost. ENC-SDPT3 spends lots of time on Human Gene and Movie datasets, but it achieves the best average results under all the metrics on these datasets. In conclusion, the SDPT3 package may occupy more computational cost, but it is more reliable than the L-BFGS optimization method. In a conclusion, our method keeps a good balance between performance and computational cost.

4.7. Optimization Methods and Convergence

It can be found that optimization method is also important for the performance of nonnegative components learning module. The objective function $T_\theta(\theta)$ with L-BFGS optimizing method is converged and we report its value in learning process in Fig. 2. It

Table 5
Computation time (in seconds) of comparing methods.

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	ENC-SDPT3
Alpha	1.22	3.82	1.61	44.24	5.21	1.16	1.05	3.67
Cdc	0.45	1.75	1.45	35.41	3.43	0.93	0.89	3.01
Cold	0.42	1.45	0.78	10.02	0.33	0.47	0.33	1.35
Diau	0.53	1.24	0.89	7.42	0.47	0.65	0.57	1.69
Dtt	0.37	1.17	0.57	8.55	0.38	0.46	0.89	1.36
Elu	0.91	1.38	1.16	20.12	3.05	0.92	0.79	3.04
Heat	0.22	0.89	0.45	13.89	0.78	0.59	0.52	1.62
Spo	0.39	1.27	0.44	11.99	0.49	0.59	0.48	1.67
Spoem	0.44	1.26	0.55	6.28	0.28	0.33	0.42	0.98
Spo5	0.33	1.08	0.89	7.91	0.23	0.39	0.75	1.12
s-JAFFE	2.73	3.46	2.48	305.45	7.65	27.16	1.52	1.77
Human Gene	76.88	35.90	36.91	778.32	92.28	150.87	58.32	231.96
Movie	43.86	75.53	46.01	6117.85	900.42	124.94	11.45	412.21

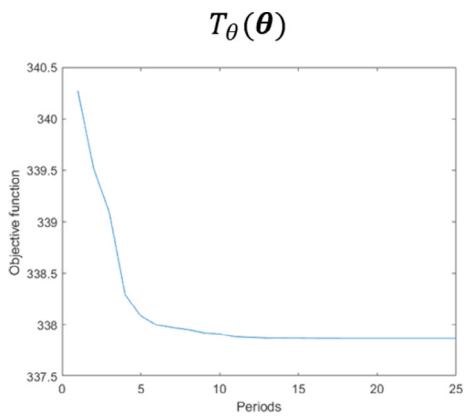


Fig. 2. The value of $T_\theta(\theta)$ in the learning process of LDL-ENC-BFGS on the sJAFFE dataset. The objective functions converges to the minimum..

is noteworthy that on Human Gene dataset, the condition number of $\mathbf{B}^T\mathbf{B}$ is very large, i.e., achieves 6.66×10^{17} in one test, which will lead to unreliable computation of inverse matrix. Thus, we compute moore–penrose inverse for the $\mathbf{B}^T\mathbf{B}$ to keep the numerical stability. Also, SDPT3 is reliable for computing components on all the datasets.

5. Conclusion

LDL is a novel learning framework considering the importance of every label for one instance, which can be seen as a generation of the SLL and MLL problems. In this paper, a novel approach is proposed to discover interpretable basis and reduce the dimensionality of feature space for solving LDL. The approach first learns the basis for each label and transforms the original input features into coefficients of their representation in the basis space. Then, a maximum entropy classification model is employed to learn label distribution on the coefficient matrix. The coefficient matrix in embedded basis space reduces complexity for the classification model on most datasets and brings better performance. Moreover, no more parameters are required in this method, which needs no additional knowledge. The experimental results on several real-world datasets demonstrate that our approach discovers good bases that help improve the performance with not too much computational cost. In summary, our method is an efficient and competitive approach and suitable for the LDL problem.

In the future, a possible direction is to explore more on the basis. Several bases for one label may represent the features better than exploiting only one basis. Another interesting issue is to provide more theoretical analysis on the learning of bases and optimization methods to validate it in theory.

CRediT authorship contribution statement

Tianyue Zhang: Conceptualization, Methodology, Software, Writing - original draft. **Yingke Mao:** Validation, Investigation, Funding acquisition. **Furao Shen:** Supervision, Project administration. **Jian Zhao:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported by the State Grid Corporation of China under project number 520950200009.

References

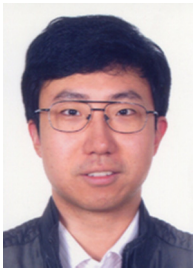
[1] O. Badarneh, M.A. Ayyoub, N. Alhindawi, L.A. Tawalbeh, Y. Jararweh, Fine-grained emotion analysis of arabic tweets: A multi-target multi-label approach, in: ICSC, IEEE Computer Society, 2018, pp. 340–345..
[2] H. Jain, Y. Prabhu, M. Varma, Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications, in: KDD, ACM, 2016, pp. 935–944..
[3] Y. Zhou, H. Xue, X. Geng, Emotion distribution recognition from facial expressions, in, ACM Multimedia, ACM (2015) 1247–1250.
[4] X. Geng, Label distribution learning, IEEE Trans. Knowl. Data Eng. 28 (7) (2016) 1734–1748.
[5] X. Geng, C. Yin, Z. Zhou, Facial age estimation by learning from label distributions, IEEE Trans. Pattern Anal. Mach. Intell. 35 (10) (2013) 2401–2412.
[6] X. Jia, W. Li, J. Liu, Y. Zhang, Label distribution learning by exploiting label correlations, AAAI, 2–7, New Orleans, Louisiana, USA, February, 2018, pp. 3310–3317.
[7] X. Zheng, X. Jia, W. Li, Label distribution learning by exploiting sample correlations locally, in: AAAI, New Orleans, Louisiana, USA, February 2-7, 2018, 2018, pp. 4556–4563..
[8] X. Jia, Z. Li, X. Zheng, W. Li, S. Huang, Label distribution learning with label correlations on local samples, IEEE Trans. Knowl. Data Eng. 33 (4) (2021) 1619–1631.
[9] P. Zhao, Z. Zhou, Label distribution learning by optimal transport, in: AAAI, New Orleans, Louisiana, USA, February 2-7, 2018, 2018, pp. 4506–4513..
[10] A. Zychowski, J. Mandziuk, Duo-ldl method for label distribution learning based on pairwise class dependencies, Appl. Soft Comput. 110 (2021) 107585.
[11] X. Geng, Q. Wang, Y. Xia, Facial age estimation by adaptive label distribution learning, in: ICPR, IEEE Computer Society, 2014, pp. 4465–4470..
[12] J. Wang, X. Geng, Label distribution learning machine, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18–24 July 2021, Virtual Event, Vol. 139 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 10749–10759..
[13] X. Geng, K. Smith-Miles, Z. Zhou, Facial age estimation by learning from label distributions, in: AAAI, AAAI Press, 2010..
[14] B. Gao, H. Zhou, J. Wu, X. Geng, Age estimation using expectation of label distribution learning, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden., 2018, pp. 712–718..
[15] H. Zhang, Y. Zhang, X. Geng, Practical age estimation using deep label distribution learning, Frontiers Comput. Sci. 15 (3) (2021) 153318.
[16] K. Wang, X. Geng, Binary coding based label distribution learning, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden., 2018, pp. 2783–2789..
[17] Z. Zhang, C. Lai, H. Liu, Y. Li, Infrared facial expression recognition via gaussian-based label distribution learning in the dark illumination environment for human emotion detection, Neurocomputing 409 (2020) 341–350.
[18] Z. Liu, Z. Chen, J. Bai, S. Li, S. Lian, Facial pose estimation by deep learning from label distributions, in: 2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27–28, 2019, IEEE, 2019, pp. 1232–1240..
[19] T. Liu, J. Wang, B. Yang, X. Wang, Ngdnet: Nonuniform gaussian-label distribution learning for infrared head pose estimation and on-task behavior understanding in the classroom, Neurocomputing 436 (2021) 210–220.
[20] M. Ling, X. Geng, Soft video parsing by label distribution learning, Frontiers Comput. Sci. 13 (2) (2019) 302–317.
[21] N. Gillis, The why and how of nonnegative matrix factorization, CoRR abs/1401.5226..
[22] C.H.Q. Ding, T. Li, M.I. Jordan, Convex and semi-nonnegative matrix factorizations, IEEE Trans. Pattern Anal. Mach. Intell. 32 (1) (2010) 45–55.
[23] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, B.W. Schuller, A deep semi-nmf model for learning hidden representations, in: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, Vol. 32 of JMLR Workshop and Conference Proceedings, 2014, pp. 1692–1700..
[24] T. Takahashi, T. Hori, C.M. Wilk, S. Sagayama, Semi-supervised NMF in the chroma domain applied to music harmony estimation, in: APSIPA, IEEE, 2018, pp. 1636–1641..
[25] X. Jia, F. Sun, H. Li, Y. Cao, X. Zhang, Image multi-label annotation based on supervised nonnegative matrix factorization with new matching measurement, Neurocomputing 219 (2017) 518–525.
[26] M H Van Benthem, M R Keenan, Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems[J], Journal of Chemometrics: A Journal of the Chemometrics Society 18 (10) (2004) 441–450.
[27] Y.-X. Yuan, A modified bfgs algorithm for unconstrained optimization, IMA Journal of Numerical Analysis 11 (3) (1991) 325–332.
[28] R.H. Tütüncü, K. Toh, M.J. Todd, Solving semidefinite-quadratic-linear programs using SDPT3, Math. Program. 95 (2) (2003) 189–217.

[29] I. CVX Research, CVX: Matlab software for disciplined convex programming, version 2.0, URL:http://cvxr.com/cvx (2012)..
[30] M. Grant, S. Boyd, Graph implementations for nonsmooth convex programs, in: V. Blondel, S. Boyd, H. Kimura (Eds.), Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 2008, pp. 95–110, URL:http://stanford.edu/ boyd/graph_dcp.html.
[31] M.B. Eisen, P.T.Spellman, P.O.Brown, D.Botstein, Cluster analysis and display of genome-wide expression patterns, in: Proc. Nat. Acad. Sci. USA, 1998, pp. 14863–14868..
[32] M.J. Lyons, S. Akamatsu, M. Kamachi, J. Gyoba, in: Coding facial expressions with gabor wavelets, in: 3rd International Conference on Face & Gesture Recognition (FG '98), Nara, Japan, 1998, pp. 200–205.
[33] T. Ahonen, A. Hadid, M. Pietikäinen, Face description with local binary patterns: Application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 2037–2041.

Tianyue Zhang received the B.Sc. degree from Nanjing University, Nanjing, China, in 2017. She is currently pursuing the Ph.D. degree in computer science from Nanjing University, Nanjing, China. Her current research interests include online learning and incremental learning.

Yingke Mao received the B.S. degree form Tsinghua University, Beijing, China in 2003 and the Ph.D. degree from Tsinghua University, Beijing, China in 2008. Currently, he is a senior engineer of State Grid Corporation of China and the chief of a UHV substation. His research interests include insulation, flexible HVDC, online monitoring and operation maintenance.

Furao Shen received the B.Sc. and M.Sc. degrees in mathematics from Nanjing University, Nanjing, China, in 1995 and 1998, respectively, and the Ph.D. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 2006. He is currently a Full Professor of computer science and technology with Nanjing University.
His current research interests include neural computing and robotic intelligence.



Jian Zhao received the B.S. degree from Nanjing University, Nanjing, China, in 2001, the M.Sc. degree from Hamburg University of Technology, Hamburg, Germany, in 2004, and the Ph.D. degree in electrical engineering from Swiss Federal Institute of Technology (ETH) Zurich, Switzerland, in 2010. Since December 2010, he has been with the Institute for Infocomm Research, Singapore. His research interests include optimization techniques in wireless communications, multiuser MIMO communications, and cooperative communications.
Dr. Zhao has received a number of awards, including the DAAD-Siemens Asia 21st Century Scholarship, IEEE Globecom 2008 Best Paper Award, and Chinese Government Award for Outstanding Self-Financed Students Abroad.