Multi-label Classification via Feature-aware Implicit Label Space Encoding

Zijia Lin[1,2] LINZIJIA07@TSINGHUA.ORG.CN
Guiguang Ding[2] DINGGG@TSINGHUA.EDU.CN
Mingqing Hu[3] HUMINGQING@ICT.AC.CN
Jianmin Wang[2] JIMWANG@TSINGHUA.EDU.CN
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, P.R. China
[2]School of Software, Tsinghua University, Beijing, P.R. China
[3]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P.R. China

Abstract

To tackle a multi-label classification problem with many classes, recently label space dimension reduction (LSDR) is proposed. It encodes the original label space to a low-dimensional latent space and uses a decoding process for recovery. In this paper, we propose a novel method termed FaIE to perform LSDR via **F**eature-**a**ware **I**mplicit label space **E**ncoding. Unlike most previous work, the proposed FaIE makes no assumptions about the encoding process and directly learns a code matrix, *i.e.* the encoding result of some implicit encoding function, and a linear decoding matrix. To learn both matrices, FaIE jointly maximizes the recoverability of the original label space from the latent space, and the predictability of the latent space from the feature space, thus making itself feature-aware. FaIE can also be specified to learn an explicit encoding function, and extended with kernel tricks to handle non-linear correlations between the feature space and the latent space. Extensive experiments conducted on benchmark datasets well demonstrate its effectiveness.

1. Introduction

As an extension of traditional multi-class classification, multi-label classification allows associating an instance with multiple labels for describing it more completely. And it is utilized for tackling numerous real-world applications like multi-label text classification (Katakis et al., 2008), semantic image annotation (Carneiro et al., 2007) and music emotion categorization (Tsoumakas et al., 2008b), *etc*.

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

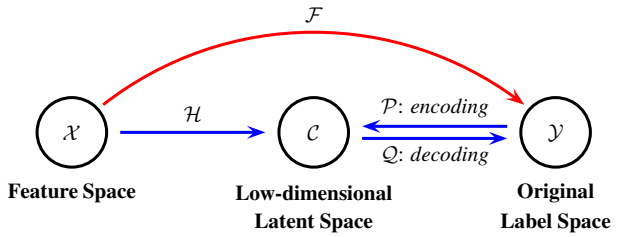


Figure 1. An illustration of the principles of traditional multi-label classification methods (red) and those with LSDR (blue).

Recently, with the prevalence of web-based applications, instances in scenarios of multi-label classification tend to be associated with labels from large vocabularies. For example, users in the picture sharing community Flickr can annotate their images with tags from millions of candidates. As revealed by Kapoor *et al.* (2012), the large vocabularies force many existing effective multi-label classification methods (Boutell et al., 2004; Evgeniou et al., 2005; Tsochantaridis et al., 2006; Tsoumakas & Katakis, 2007; Argyriou et al., 2008; Hariharan et al., 2010; Tsoumakas et al., 2010; Zhou & Tao, 2012) to become unaffordable, since generally they will learn a predictive model for each label and combine them in a certain manner for prediction.

With a large vocabulary, the number of needed predictive models will be quite large, making the training costs unaffordable. To tackle such problems, recently academia has seen efforts made to perform label space dimension reduction (LSDR) (Hsu et al., 2009; Tai & Lin, 2010; Kapoor et al., 2012; Zhou et al., 2012; Chen & Lin, 2012; Wicker et al., 2012). As illustrated in Fig. 1, for LSDR each original high-dimensional label vector is encoded to a low-dimensional code vector in a latent space. Then predictive models are trained from instance features to code vectors, whose quantity is much smaller and thus can significantly reduce the training costs. To predict an unseen instance, a low-dimensional code vector is firstly obtained with the

learnt predictive models on its features and then efficiently decoded for recovering its predicted label vector. If the learnt predictive models and the decoding process are effective enough, LSDR is expected to yield acceptable classification performance at much lower costs.

Previous researches on LSDR mostly require an explicit encoding function, *e.g.* a linear one, for mapping original label vectors to code vectors. However, since the optimal mapping can be complicated and even indescribable, assuming an explicit encoding function may not well model it. In this paper, we propose a novel method termed FaIE to perform LSDR via **F**eature-**a**ware **I**mplicit label space **E**ncoding. Specifically, FaIE directly learns a code matrix and a linear decoding matrix without any assumption concerning the encoding process but just knowing that the code matrix consisting of code vectors is the encoding result of some implicit encoding function. Compared to explicit encoding, implicit encoding can reduce the risk of using an inappropriate predefined encoding function and thus probably learn a better encoding result. To learn the code matrix and the decoding matrix, FaIE jointly maximizes the recoverability of the original label space and the predictability of the latent space, making itself feature-aware. FaIE can also be specified to learn an explicit encoding function, and extended with kernel tricks to handle non-linear correlations between the feature space and the latent space.

The remainder of this paper is organized as follows. Section 2 gives an overview of related work. Section 3 elaborates on the proposed FaIE and presents the formula details. Then experimental settings, results and analyses are given in Section 4. Finally we conclude the paper in Section 5.

2. Related Work

To tackle a multi-label classification problem with many classes (*i.e.* a large vocabulary), many effective methods were proposed, like constructing a hierarchy of multi-label classifiers (Tsoumakas et al., 2008a), refining the output of heuristic efficient classifiers (Dekel & Shamir, 2010), or performing label selection (Balasubramanian & Lebanon, 2012), *etc*. Recently, LSDR was also proposed.

To the best of our knowledge, Hsu *et al.* (2009) could be the first to focus on LSDR. Specifically, Hsu *et al.* exploited the sparsity of the original label space and proposed to linearly encode it as compressed sensing (CS) and use standard recovery algorithms like CoSaMP (Needell & Tropp, 2009) for decoding. As a further research, (Kapoor et al., 2012) considered both label space compression and predictive model learning in a single probabilistic model, and derived a Bayesian framework termed BML-CS for multi-label classification via jointly optimizing over both.

Moreover, Tai and Lin (2010) proposed to perform principle label space transformation (PLST) for seeking important correlations between labels, which is essentially PCA (Jolliffe, 1986) for the label space. Chen and Lin (2012) further proposed feature-aware conditional principal label space transformation (CPLST), considering the predictability of code vectors. Zhou *et al.* (2012) proposed to take the signs of linear Gaussian random projection results as the code vectors and utilize a series of Kullback-Leibler divergence based hypothesis tests for decoding. Wicker *et al.* (2012) proposed MLC-BMaD for LSDR via boolean matrix decomposition on the tagging matrix, factorizing it as the product of a binary code matrix and a binary decoding matrix. As a special case of linear encoding, (Bi & Kwok, 2013) proposed an efficient randomized sampling procedure for selecting a column subset of the tagging matrix that can well span it, termed ML-CSSP.

By surveying previous researches on LSDR, we realize that they mostly require an explicit encoding function for mapping original label vectors to code vectors, which, as analysed previously, may not well model the optimal mapping. MLC-BMaD seems to be the only one allowing implicit encoding, as it directly learns the code matrix without any assumption on the encoding process. However, MLC-BMaD is feature-unaware, and the learnt code matrix could be less predictable, which will probably lower the prediction performance. Hence in this paper we propose FaIE, which not only allows implicit encoding, but also is feature-aware.

3. Proposed Approach

3.1. Preliminaries

Given a set of N training instances with corresponding labels $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^F$ and $\mathbf{y}_i \in \mathcal{Y} \subset \{0,1\}^K$ are respectively the F-dimensional feature vector and K-dimensional label vector of the ith instance, multi-label classification is to learn the mapping $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Y}$ for predicting the label vector of any unseen instance based on its features, as illustrated in Fig. 1. Here each entry of the label vector of an instance indicates whether it belongs to a certain class (1) or not (0). Feature vectors and label vectors of the training data respectively form the feature matrix $X_{N\times F}$ and the tagging matrix $Y_{N\times K}$ row by row.

Many existing effective multi-label classification methods derive $\mathcal{F}$ by learning K binary mappings $\{\mathcal{F}_i | \mathcal{F}_i : \mathcal{X} \mapsto \mathcal{Y}_i, i = 1, 2, \ldots, K\}$, where $\mathcal{Y}_i \subset \{0,1\}$ is the ith dimension of the original label space corresponding to the ith class. When the number of classes (*i.e.* K) becomes large, the training costs of those methods would be unaffordable. To tackle the challenge, recently LSDR was proposed. Then learning $\mathcal{F}$ would be transformed into firstly encoding each original label vector to a low-dimensional code vector in a latent space $\mathcal{C} \subset \mathbb{R}^L (L \ll K)$ with an

encoding process $\mathcal{P}: \mathcal{Y} \mapsto \mathcal{C}$ and then learning a mapping $\mathcal{H}: \mathcal{X} \mapsto \mathcal{C}$ *w.r.t* the code vectors, which can be derived by learning L mappings $\{\mathcal{H}_i | \mathcal{H}_i : \mathcal{X} \mapsto \mathcal{C}_i, i = 1, 2, \ldots, L\}$ with $\mathcal{C}_i \subset \mathbb{R}$ being the ith dimension of $\mathcal{C}$. And the predicting process for an unseen instance will be transformed into firstly predicting an L-dimensional code vector in $\mathcal{C}$ with $\mathcal{H}$ on its features and then recovering a K-dimensional predicted label vector with a decoding process $\mathcal{Q}: \mathcal{C} \mapsto \mathcal{Y}$, as illustrated in Fig. 1. Note that for LSDR here $\mathcal{C}$ is derived from $\mathcal{Y}$ rather than $\mathcal{X}$, via explicit encoding like linear encoding or implicit encoding such as matrix decomposition, and generally the decoding process $\mathcal{Q}$ needs to be specified before deriving $\mathcal{C}$, while the mapping $\mathcal{H}$ is unspecified and will be open for any mapping algorithm after $\mathcal{C}$ is derived.

3.2. Proposed FaIE

As mentioned previously, FaIE makes no assumptions concerning the encoding process $\mathcal{P}$, and it directly learns a code matrix $C_{N\times L}$ consisting of code vectors and a linear decoding matrix $D_{L\times K}$ by decomposing the tagging matrix $Y_{N\times K}$ as the product of both, *i.e.* $Y \sim CD$.

Generally, the classification performance of LSDR methods depends on both the predictive mapping $\mathcal{H}$ and the decoding process $\mathcal{Q}$. Therefore, it would be important for code vectors to be predictable, having a strong correlation with instance features, as revealed by (Zhang & Schneider, 2011). And the original label vectors should also be highly recoverable via decoding the code vectors. Then to better learn C and D, FaIE jointly maximizes the recoverability of the original label space and the predictability of the latent space. Denoting the former as $\Phi(Y, C, D)$ and the latter as $\Psi(X, C)$, its objective function is as follows.

$$\Omega = \max_{C,D} \ \Phi(Y, C, D) + \alpha\Psi(X, C) \tag{1}$$

where α is a parameter for balancing recoverability and predictability. Actually, FaIE derives C and D via matrix decomposition on Y, and utilizes $\Psi(X, C)$, which considers correlations between X and C, as side information to make C feature-aware. As will be detailed later, given C, the optimal D can be derived as a closed-form expression. Then for model simplicity, the objective function above is reformulated to be the following one concerning only C.

$$\Omega = \max_{C} \ \Phi(Y, C) + \alpha\Psi(X, C) \tag{2}$$

3.2.1. Recoverability of Original Label Space

To improve the recoverability of the original label space, the difference between the tagging matrix Y and the recovered one using the code matrix C and the decoding matrix D, denoted as $\mathcal{E}(Y, C, D)$, is expected to be minimized.

$$\mathcal{E}(Y, C, D) = \|Y - CD\|_{fro}^2 \tag{3}$$

where $\|\cdot\|_{fro}$ is the *Frobenius norm* of a matrix. Given C, the optimal D to minimize $\mathcal{E}(Y, C, D)$ can be derived as the following closed-form expression by solving $\frac{\partial \mathcal{E}}{\partial D} = 0$.

$$D = (C^T C)^{-1} C^T Y \tag{4}$$

To avoid redundant information in the latent space and then enable the proposed FaIE to encode the original label space more compactly, we assume that dimensions of C are uncorrelated and thus orthonormal, as shown in formula (5) .

$$C^T C = I_{L\times L} \tag{5}$$

where $I_{L\times L}$ is an identity matrix. Then the optimal $D = C^T Y$, and formula (3) can be reformulated as follows.

$$\mathcal{E}(Y, C, D) = \mathbf{Tr}[Y^T Y - Y^T C C^T Y] \tag{6}$$

where $\mathbf{Tr}[\cdot]$ denotes the *trace* of a matrix. With $\mathbf{Tr}[Y^T Y]$ being a constant, minimizing $\mathcal{E}(Y, C, D)$ is equivalent to maximizing $\mathbf{Tr}[Y^T C C^T Y]$, which can be seen as an expression of the recoverability of the original label space, *i.e.* $\Phi(Y, C)$. And thus we can derive the following formula.

$$\begin{array}{ll} \Phi(Y, C) = & \mathbf{Tr}[Y^T C C^T Y] = \mathbf{Tr}[C^T Y Y^T C] \\ s.t. & C^T C = I \end{array} \tag{7}$$

3.2.2. Predictability of Latent Space

As revealed in (Zhang & Schneider, 2011), to improve the predictability of the low-dimensional latent space, the code matrix C is supposed to be strongly correlated with instance features. Here we firstly consider linear correlations, and will handle non-linear ones with kernel tricks later. Considering a linear projection $\mathbf{r}$ for the feature space and a dimension $\mathbf{c}$ of the latent space, *i.e.* a column of C, the correlation between features and $\mathbf{c}$, *i.e.* $\rho(X, \mathbf{c})$, can be defined as follows.

$$\rho(X, \mathbf{c}) = \frac{(X\mathbf{r})^T \mathbf{c}}{\sqrt{(X\mathbf{r})^T (X\mathbf{r})}\sqrt{\mathbf{c}^T \mathbf{c}}} \tag{8}$$

Note that here the definition of $\rho(X, \mathbf{c})$ makes no assumptions about the encoding process from Y to $\mathbf{c}$. According to the orthonormality constraint for C, *i.e.* formula (5), $\mathbf{c}^T \mathbf{c} = 1$ will hold for any column of C. Then to maximize $\rho(X, \mathbf{c})$, we introduce the following optimization problem.

$$\max \ (X\mathbf{r})^T \mathbf{c} \quad s.t. \quad (X\mathbf{r})^T X\mathbf{r} = 1 \tag{9}$$

When given a dimension $\mathbf{c}$ of the latent space, the maximal $(X\mathbf{r})^T \mathbf{c}$ reflects its potential maximal correlation with the feature space, and thus the maximal $(X\mathbf{r})^T \mathbf{c}$ can be seen as an expression of the predictability of $\mathbf{c}$. Specifically, with $\mathbf{c}$ fixed, the optimal $\mathbf{r}$ for formula (9), denoted as $\mathbf{r}^*$, can be derived as follows with the method of Lagrange multipliers.

$$\mathbf{r}^* = \frac{(X^T X)^{-1} X^T \mathbf{c}}{\sqrt{\mathbf{c}^T X (X^T X)^{-1} X^T \mathbf{c}}}$$

Note that here we assume X^TX to be invertible, which can be ensured by adding a tiny value to entries on the diagonal. Then the predictability of $\mathbf{c}$, denoted as $\psi(X, \mathbf{c})$, can be derived as follows by substituting $\mathbf{r}^*$ into formula (9).

$$\psi(X, \mathbf{c}) = (X\mathbf{r}^*)^T\mathbf{c} = \sqrt{\mathbf{c}^T\Delta\mathbf{c}} \tag{10}$$

where $\Delta = X(X^TX)^{-1}X^T$. Therefore, to improve the predictability of the latent space, each column $\mathbf{c}$ of the code matrix C is supposed to maximize $\psi(X, \mathbf{c})$. As maximizing $\psi(X, \mathbf{c})$ can be guaranteed by maximizing $\mathbf{c}^T\Delta\mathbf{c}$, the overall predictability of the code matrix C is as follows.

$$\begin{array}{ll} \Psi(X, C) = & \sum_i C_{\cdot,i}^T \Delta C_{\cdot,i} = \mathbf{Tr}[C^T\Delta C] \\ s.t. & C^TC = I \end{array} \tag{11}$$

where $C_{\cdot,i}$ is the ith column of C. Based on the formulas above regarding predictability of the latent space, we can derive the following lemma.

Lemma 1. *For any given matrix $C_{N\times L}$ satisfying $C^TC = I$, $\mathbf{Tr}[C^T\Delta C]$ has an upper bound being $\min(L, rank(\Delta))$.*

For detailed demonstrations, one can refer to the supplementary appendix. The lemma could be helpful for parameter tuning of FaIE, *i.e.* α in formula (2). Because generally α need not be further increased when $\mathbf{Tr}[C^T\Delta C]$ is observed to converge to its upper bound, otherwise the performance of FaIE will probably go down due to the loss of recoverability, as will be validated by our experiments.

3.2.3. Solution and Implementation Issues

The objective function of the proposed FaIE, *i.e.* formula (2) , can be detailed as follows.

$$\begin{array}{lll} \Omega & = & \max_C \ \mathbf{Tr}[C^TYY^TC] + \alpha\mathbf{Tr}[C^T\Delta C] \\ & = & \max_C \ \mathbf{Tr}[C^T(YY^T + \alpha\Delta)C] \\ s.t. & & C^TC = I \end{array} \tag{12}$$

where $\Delta = X(X^TX)^{-1}X^T$. Then for optimization, any column $C_{\cdot,i}$ of the code matrix C can be derived with the following optimization sub-problem.

$$\begin{array}{ll} \Omega_i = & \max_{C_{\cdot,i}} \ C_{\cdot,i}^T(YY^T + \alpha\Delta)C_{\cdot,i} \\ s.t. & C_{\cdot,i}^TC_{\cdot,i} = 1, C_{\cdot,j}^TC_{\cdot,i} = 0 \ (\forall j < i) \end{array} \tag{13}$$

With the method of Lagrange multipliers, the optimal $C_{\cdot,i}$ should satisfy the following optimality condition.

$$(YY^T + \alpha\Delta)C_{\cdot,i} = \lambda_i C_{\cdot,i} \tag{14}$$

where λ_i is the introduced Lagrange multiplier and will also be the optimal value of the sub-problem. It can be seen that the optimization for C can be transformed to an eigenvalue problem. And the normalized eigenvectors of $(YY^T + \alpha\Delta)$ corresponding to the *top* L largest eigenvalues will form the optimal code matrix C. The computational complexity for $(YY^T + \alpha\Delta)$ is at most $\mathcal{O}(N^2K + N^2F + 2F^2N + F^3)$. And for the eigenvalue problem, as L is generally much smaller than N, it can be solved efficiently with iterative methods like Arnoldi iteration (Lehoucq & Sorensen, 1996) which can achieve an optimal computational complexity of $\mathcal{O}(L^2N)$. Then with C, predictive models from feature space to the latent space can be learnt to predict code vectors of unseen instances. And the predicted label vectors can be recovered using the decoding matrix $D = C^TY$. An illustration of the implementation of FaIE is given in Algorithm 1.

Algorithm 1 Implementation of FaIE

Input: Feature matrix X_{tr} and tagging matrix Y_{tr} of training set, feature matrix X_{ts} of test test, predefined parameter α and target dimension L of the latent space
Output: Predicted tagging matrix Y_{ts} of test set

$\Delta = X_{tr}(X_{tr}^TX_{tr})^{-1}X_{tr}^T$
$\Omega = Y_{tr}Y_{tr}^T + \alpha\Delta$
$C_{tr} = eigenvector(\Omega, L)$ {normalized eigenvectors of Ω corresponding to the top L largest eigenvalues}
$D = C_{tr}^TY_{tr}$ {decoding matrix}
learn predictive models: $\mathcal{H}(X_{tr}) \rightarrow C_{tr}$
$C_{ts} = \mathcal{H}(X_{ts})$ {predicted code vectors of test set}
$Y_{ts} = round(C_{ts}D)$

3.3. Explicit Encoding: a Linear Encoding Case

Though the proposed FaIE makes no assumptions concerning the encoding process, it can also be specified to learn an explicit encoding function as most previous work, provided that the function is optimizable, *e.g.* a linear one.

In the special case of linear encoding where an explicit linear encoding matrix $P_{K\times L}$ is required, the code matrix C can be expressed as $C = YP$. Then by replacing C with YP in the objective function of FaIE, *i.e.* formula (12), we can derive the following one for linear encoding.

$$\begin{array}{ll} \Omega = & \max_P \ \mathbf{Tr}[P^T(Y^TYY^TY + \alpha Y^T\Delta Y)P] \\ s.t. & P^TY^TYP = I \end{array} \tag{15}$$

Similarly, by decomposing Ω into L optimization sub-problems *w.r.t* each column of P and using the method of Lagrange multipliers, we derive that each column $P_{\cdot,i}$ of the optimal P should satisfy the following condition.

$$(Y^TYY^TY + \alpha Y^T\Delta Y)P_{\cdot,i} = \lambda_i(Y^TY)P_{\cdot,i} \tag{16}$$

where λ_i is the introduced Lagrange multiplier and will also be the optimal value of the optimization sub-problem *w.r.t* $P_{\cdot,i}$. It can be seen that the optimization for P can be transformed to a general eigenvalue problem. Specifically, the normalized eigenvectors corresponding to the *top*

L largest eigenvalues will form the optimal P. And the corresponding linear decoding matrix is $D = (YP)^T Y$.

3.4. Kernel version

With kernel tricks, the proposed FaIE can be extended to handle non-linear correlations between the feature space and the latent space to measure the predictability of the code matrix, which is termed *kernel*-FaIE.

In *kernel*-FaIE, each feature vector $\mathbf{x}_i$ is mapped to the Reproducing Kernel Hilbert Space (RKHS) as θ_i, which also forms a kernel feature matrix Θ row by row. In RKHS, the inner product $\langle \theta_i, \theta_j \rangle$ between θ_i and θ_j will be equal to $\kappa(\mathbf{x}_i, \mathbf{x}_j)$, where $\kappa(\cdot, \cdot)$ is the introduced kernel function. And then with a non-linear kernel function, linear correlations between the RKHS and the latent space will reflect the non-linear correlations between the original feature space and the latent space. Similar to formula (8), we can consider a linear projection $\mathbf{r}'$ for kernel features in RKHS and a column $\mathbf{c}$ of the code matrix to measure the correlation $\rho(\Theta, \mathbf{c})$. Like kernel CCA (Hardoon et al., 2004), we require $\mathbf{r}'$ to be in the span of the kernel feature vectors, *i.e.* $\mathbf{r}' = \Theta^T \beta$ where β is a N-dimensional weighting vector. Then $\rho(\Theta, \mathbf{c})$ can be measured as follows.

$$\begin{aligned} \rho(\Theta, \mathbf{c}) &= \frac{(\Theta\Theta^T\beta)^T\mathbf{c}}{\sqrt{(\Theta\Theta^T\beta)^T(\Theta\Theta^T\beta)}\sqrt{\mathbf{c}^T\mathbf{c}}} \\ &= \frac{(K\beta)^T\mathbf{c}}{\sqrt{(K\beta)^T(K\beta)}\sqrt{\mathbf{c}^T\mathbf{c}}} \end{aligned} \tag{17}$$

where $K = \Theta\Theta^T$ is the kernel matrix. With similar derivations to subsection 3.2.2, the predictability of $\mathbf{c}$ based on non-linear correlations can be measured as $\psi'(X, \mathbf{c}) = \sqrt{\mathbf{c}^T \Gamma \mathbf{c}}$ with $\Gamma = K(K^T K)^{-1} K^T$. And then the objective function of *kernel*-FaIE is as follows.

$$\begin{aligned} \Omega = &\ \max_C \ \mathbf{Tr}[C^T(YY^T + \alpha\Gamma)C] \\ s.t. &\ \ C^T C = I \end{aligned} \tag{18}$$

It can also be transformed to an eigenvalue problem. And the corresponding linear decoding matrix is $D = C^T Y$.

3.5. Relations to previous work and discussions

With the mean values of label vectors shifted as zeros, the proposed FaIE will degenerate to PLST (Tai & Lin, 2010) when considering only the recoverability of the original label space (*i.e.* $\alpha = 0$ in formula (12)). Specifically, in this case, the objective function of FaIE is as follows.

$$\Omega = \max_C \ \mathbf{Tr}[C^T Y Y^T C], \quad s.t. \quad C^T C = I \tag{19}$$

The code matrix C consists of the normalized eigenvectors of YY^T corresponding to the *top* L largest eigenvalues, with the decoding matrix being $C^T Y$. As for PLST, its linear encoding matrix P is formed with normalized eigenvectors of $Y^T Y$ corresponding to the *top* L largest eigenvalues, with the encoding result and the decoding matrix respectively being YP and P^T. Note that YY^T and $Y^T Y$ are positive semi-definite and share the same positive eigenvalues. Provided that λ_i is the ith largest eigenvalue, we can derive that: 1) $Y^T Y P_{\cdot,i} = \lambda_i P_{\cdot,i}$; 2) $YY^T C_{\cdot,i} = \lambda_i C_{\cdot,i}$; 3) $(YY^T)[YP]_{\cdot,i} = Y(Y^T Y P_{\cdot,i}) = \lambda_i [YP]_{\cdot,i}$; 4) $(Y^T Y)[Y^T C]_{\cdot,i} = Y^T(YY^T C_{\cdot,i}) = \lambda_i [Y^T C]_{\cdot,i}$. Then for FaIE and PLST, we can find one-to-one correspondences between the ith columns of their encoding results (*i.e.* $C_{\cdot,i} = \frac{[YP]_{\cdot,i}}{\sqrt{\lambda_i}}$), and between the ith rows of their decoding matrices (*i.e.* $[C^T Y]_{i,\cdot} = \sqrt{\lambda_i}[P^T]_{i,\cdot}$). Therefore, when $\alpha = 0$, FaIE is equivalent to PLST, with $C(C^T Y) = (YP)P^T$. Yet when $\alpha > 0$, C will be associated to instance features, and the encoding process from Y to C will become complicated and even indescribable.

In the case of explicit linear encoding, *i.e.* formula (15), with the mean values of label vectors and feature vectors shifted as zeros, FaIE has a close connection to CPLST (Chen & Lin, 2012) if it considers only the predictability of the latent space. The corresponding objective function is given as follows.

$$\Omega = \max_P \ \mathbf{Tr}[P^T Y^T \Delta Y P], \ s.t. \ P^T Y^T Y P = I \tag{20}$$

Meanwhile, the objective function of CPLST is as follows.

$$\Omega_1 = \max_P \ \mathbf{Tr}[P^T Y^T H Y P], \quad s.t. \quad P^T P = I \tag{21}$$

where $H = X(X^T X)^{-1} X^T = \Delta$. And thus it can be seen that in this case FaIE and CPLST will share an identical objective function but with different constraints, the former requiring dimensions of the encoding result to be orthonormal while the latter requiring dimensions of the linear encoding matrix to be orthonormal.

Another interesting observation regarding FaIE is that when the predictability of the latent space is overemphasized with an assumption that the code matrix can be directly expressed by the feature matrix, *i.e.* $C = XR$ where $R_{F \times L}$ is a regression matrix, FaIE will perform dimension reduction for both the original label space and the feature space. Specifically, with $C = XR$, the objective function of FaIE, *i.e.* formula (12), will have a constant value of predictability, and it can be simplified as follows.

$$\begin{aligned} \Omega = &\ \max_R \ \mathbf{Tr}[R^T X^T Y Y^T X R] \\ s.t. &\ \ R^T X^T X R = I \end{aligned} \tag{22}$$

Here the optimization for R can also be transformed to a general eigenvalue problem, *i.e.* $(X^T Y Y^T X) R_{\cdot,i} = \lambda_i (X^T X) R_{\cdot,i}$, but it requires that $L \leq F$. Then C can be seen as the dimension reduction result from label space with an implicit encoding function, or the linear dimension reduction result from feature space with R.

Table 1. Statistics of datasets

	domain	instances	labels	features
delicious	text	5,000	983	500
CAL500	music	502	174	68
mediamill	video	5,000	101	120
ESPGame	image	5,000	1,932	516

4. Experiments

4.1. Experimental settings

To validate the proposed FaIE, we download three benchmark datasets in different domains with relatively large vocabularies from Mulan (Tsoumakas et al., 2011) for experiments, *i.e. delicious*, *CAL500*, and *mediamill*. For reducing computational costs, here we randomly take a subset of *delicious* and *mediamill* with 5,000 instances. Moreover, following (Hsu et al., 2009), we also conduct experiments on a randomly selected subset of the image dataset *ESPGame*, and take those tags appearing at least twice in the subset to form a much larger vocabulary. Each instance in *ESPGame* is represented with a 516-D feature vector[1] extracted with Lire (Lux & Chatzichristofis, 2008). Some statistics of the datasets are given in Table 1.

For performance comparison, we take Binary Relevance (BR) (Fürnkranz et al., 2008), CS (Hsu et al., 2009), PLST (Tai & Lin, 2010), CPLST and *kernel*-CPLST (Chen & Lin, 2012), MLC-BMaD (Wicker et al., 2012) and ML-CSSP (Bi & Kwok, 2013) as baselines. BR is a widely-used multi-label classification method that trains a separate binary relevance model for each label. In our experiments, we respectively use linear SVM (L-SVM) (Fan et al., 2008) and linear ridge regression (L-RR) for BR, with the latter using 0.5 as a threshold to decide the binary (0 or 1) classification results. Note that BR does not perform LSDR and thus its performance can be a reference for other algorithms. BML-CS (Kapoor et al., 2012) is not included since it is sophisticated with numerous parameters to tune.

For FaIE, we evaluate the following variants: 1) FaIE considering only recoverability, *i.e.* formula (19), termed R-FaIE; 2) FaIE considering only predictability for linear encoding, *i.e.* formula (20), termed P-LinearFaIE; 3) FaIE over-emphasizing predictability, *i.e.* formula (22), termed OP-FaIE; 4) FaIE for explicit linear encoding, *i.e.* formula (15), termed LinearFaIE; 5) kernel version of FaIE, *i.e.* formula (18), termed *kernel*-FaIE. Note that R-FaIE and P-LinearFaIE are respectively related to PLST and CPLST.

In our experiments, each dataset is evenly and randomly divided into 5 parts. And then we perform 5 runs for each algorithm on it, taking one part for test and the rest for training in each run without duplication. Experimental results are measured with widely-used metrics in the field of multi-label classification, *i.e. label-based macroF1* and *example-based accuracy* (Zhang & Zhou, 2013), and then averaged over the 5 runs. Moreover, for each run of any algorithm, we also conduct 5-fold cross validation on the training set for selecting model parameters via grid search in predefined value ranges. Specifically, α in the proposed FaIE is selected from $\{10^{-1}, 10^{0}, \ldots, 10^{4}\}$, τ for MLC-BMaD is chosen from $\{0.1, 0.2, \ldots, 1.0\}$, and the predefined sparsity level in CS is selected from $\{1, 2, \ldots, M\}$ with M being the maximal number of labels in an instance, *etc*. Following most previous work, we utilize linear ridge regression to learn predictive models from instance features to code vectors. As for *kernel*-CPLST and *kernel*-FaIE, we empirically utilize the Gaussian kernel function and set the smoothing parameter σ as twice the mean Euclidean distance between feature vectors for each dataset. Accordingly, we utilize kernel ridge regression for both to learn the non-linear predictive models. To decide the binary (0 or 1) predicted results of multi-label classification, we round each continuous entry of the decoded label vector to 0 or 1 with a threshold of 0.5, as PLST and CPLST.

4.2. Experimental results of LSDR

We perform all algorithms on the datasets with different values of L/K (mostly from 10% to 50%) where L and K are respectively the dimension of the latent space and the original label space, as shown in Table 2, 3, 4 and 5. Note that for *ESPGame*, L/K is varied from 5% to 25%, as it has a much larger vocabulary. Experimental results with standard errors are given in the supplementary appendix.

From the experimental results, we can draw the following observations. 1) The proposed FaIE and its linear encoding case LinearFaIE generally outperform other baselines on each dataset, which well demonstrates their effectiveness. 2) FaIE outperforms LinearFaIE on all datasets, well demonstrating the superiority of implicit encoding over explicit encoding. 3) FaIE outperforms R-FaIE and LinearFaIE outperforms P-LinearFaIE, which respectively validates the importance of predictability and recoverability, and presents the superiority of jointly considering both. 4) OP-FaIE yields inferior performance to FaIE and even cannot perform LSDR for *CAL500* when $L/K \geq 40\%$ as the dimension of feature space is smaller than L, which points out the weakness of OP-FaIE and validates the superiority of keeping a good trade-off between recoverability and predictability. 5) R-FaIE yields nearly the same performance as PLST, as predicted by our theoretical analysis about their equivalence. 6) With an identical objective function but different orthogonality constraints, P-LinearFaIE seems to be slightly superior to CPLST, which validates the reasonableness of assuming the dimensions of the code matrix to be orthonormal. 7) *kernel*-FaIE outperforms FaIE on

[1] 516-D feature vector: 60-D Gabor, 192-D FCTH, 80-D Edge Histogram, 120-D Color Layout and 64-D RGB Color Histogram

Table 2. Experimental results on *delicious*

		label-based macroF1					*example-based Accuracy*				
L/K		10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
BR	L-SVM	0.0790					0.1419				
	L-RR	0.0308					0.0810				
CS		0.0057	0.0172	0.0341	0.0366	0.0320	0.0260	0.0534	0.0854	0.0932	0.0927
PLST		0.0196	0.0214	0.0217	0.0217	0.0217	0.0734	0.0762	0.0768	0.0769	0.0768
CPLST		0.0240	0.0244	0.0244	0.0244	0.0244	0.0787	0.0787	0.0788	0.0788	0.0788
MLC-BMaD		0.0180	0.0214	0.0265	0.0277	0.0288	0.0525	0.0682	0.0690	0.0703	0.0705
ML-CSSP		0.0139	0.0197	0.0230	0.0253	0.0266	0.0555	0.0659	0.0724	0.0729	0.0760
R-FaIE ($\simeq$PLST)		0.0197	0.0213	0.0216	0.0215	0.0216	0.0736	0.0764	0.0771	0.0772	0.0771
P-LinearFaIE ($\sim$CPLST)		0.0369	0.0407	0.0412	0.0412	0.0412	0.0892	0.0971	0.0987	0.0987	0.0987
OP-FaIE		0.0472	0.0515	0.0526	0.0527	0.0514	0.1073	0.1083	0.1084	0.1082	0.1083
LinearFaIE		0.0411	0.0431	0.0434	0.0435	0.0435	0.0984	0.1058	0.1061	0.1061	0.1061
FaIE		**0.0544**	**0.0591**	**0.0602**	**0.0603**	**0.0586**	**0.1198**	**0.1207**	**0.1203**	**0.1202**	**0.1116**
kernel-CPLST		0.0341	0.0341	0.0341	0.0341	0.0341	0.1048	0.1048	0.1048	0.1048	0.1048
kernel-FaIE		**0.0566**	**0.0688**	**0.0726**	**0.0744**	**0.0750**	**0.1448**	**0.1496**	**0.1506**	**0.1507**	**0.1508**

Table 3. Experimental results on *CAL500*

		label-based macroF1					*example-based Accuracy*				
L/K		10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
BR	L-SVM	0.1397					0.2436				
	L-RR	0.0569					0.1995				
CS		0.0677	0.0820	0.0906	0.0976	0.1142	0.1130	0.1299	0.1626	0.1904	0.1835
PLST		0.0604	0.0605	0.0606	0.0609	0.0608	0.2099	0.2103	0.2103	0.2106	0.2104
CPLST		0.0640	0.0643	0.0644	0.0645	0.0645	0.2003	0.2007	0.2009	0.2010	0.2010
MLC-BMaD		0.0485	0.0444	0.0420	0.0472	0.0468	0.1286	0.1215	0.1194	0.1244	0.1255
ML-CSSP		0.0453	0.0498	0.0507	0.0528	0.0543	0.1806	0.1880	0.1913	0.1958	0.1966
R-FaIE ($\simeq$PLST)		0.0596	0.0600	0.0600	0.0601	0.0600	0.2099	0.2109	0.2109	0.2109	0.2106
P-LinearFaIE ($\sim$CPLST)		0.0800	0.0976	0.1000	0.0998	0.0998	0.2093	0.2210	0.2205	0.2206	0.2206
OP-FaIE		0.1034	0.1080	0.1088	-	-	0.2283	0.2262	0.2251	-	-
LinearFaIE		0.1062	0.1107	0.1105	0.1105	0.1105	0.2329	0.2300	0.2301	0.2299	0.2300
FaIE		**0.1199**	**0.1245**	**0.1260**	**0.1249**	**0.1245**	**0.2413**	**0.2414**	**0.2417**	**0.2384**	**0.2379**
kernel-CPLST		0.0754	0.0774	0.0774	0.0774	0.0774	0.2139	0.2148	0.2148	0.2148	0.2148
kernel-FaIE		**0.1178**	**0.1243**	**0.1250**	**0.1290**	**0.1291**	**0.2429**	**0.2443**	**0.2422**	**0.2430**	**0.2419**

Table 4. Experimental results on *mediamill*

		label-based macroF1					*example-based Accuracy*				
L/K		10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
BR	L-SVM	0.0554					0.3130				
	L-RR	0.0454					0.4173				
CS		0.0056	0.0145	0.0134	0.0311	0.0274	0.0103	0.0296	0.0343	0.1403	0.1357
PLST		0.0419	0.0435	0.0436	0.0436	0.0436	0.4117	0.4144	0.4142	0.4141	0.4143
CPLST		0.0433	0.0440	0.0440	0.0440	0.0440	0.4142	0.4148	0.4149	0.4149	0.4148
MLC-BMaD		0.0412	0.0426	0.0425	0.0423	0.0425	0.4027	0.4037	0.4027	0.4027	0.4027
ML-CSSP		0.0343	0.0406	0.0428	0.0416	0.0437	0.3378	0.3954	0.4058	0.4045	0.4148
R-FaIE ($\simeq$PLST)		0.0419	0.0438	0.0441	0.0440	0.0440	0.4121	0.4150	0.4150	0.4150	0.4151
P-LinearFaIE ($\sim$CPLST)		0.0425	0.0444	0.0453	0.0452	0.0452	0.4135	0.4145	0.4151	0.4155	0.4156
OP-FaIE		0.0450	0.0469	0.0472	0.0474	0.0471	0.4163	0.4185	0.4182	0.4177	0.4179
LinearFaIE		0.0444	0.0463	0.0464	0.0461	0.0461	0.4154	0.4172	0.4160	0.4153	0.4154
FaIE		**0.0570**	**0.0595**	**0.0609**	**0.0602**	**0.0607**	**0.4327**	**0.4339**	**0.4338**	**0.4334**	**0.4336**
kernel-CPLST		0.0492	0.0521	0.0521	0.0521	0.0521	0.4228	0.4234	0.4240	0.4240	0.4240
kernel-FaIE		**0.0682**	**0.0805**	**0.0852**	**0.0857**	**0.0855**	**0.4401**	**0.4440**	**0.4447**	**0.4423**	**0.4385**

Table 5. Experimental results on *ESPGame*

		label-based macroF1					*example-based Accuracy*				
L/K		5%	10%	15%	20%	25%	5%	10%	15%	20%	25%
BR	L-SVM	0.0213					0.0726				
	L-RR	0.0014					0.0452				
CS		0.0004	0.0006	0.0013	0.0013	0.0018	0.0166	0.0213	0.0445	0.0467	0.0505
PLST		0.0008	0.0008	0.0008	0.0008	0.0008	0.0457	0.0457	0.0457	0.0457	0.0457
CPLST		0.0008	0.0008	0.0008	0.0008	0.0008	0.0469	0.0470	0.0470	0.0470	0.0470
MLC-BMaD		0.0009	0.0009	0.0010	0.0009	0.0010	0.0450	0.0449	0.0458	0.0446	0.0446
ML-CSSP		0.0008	0.0005	0.0007	0.0007	0.0008	0.0432	0.0280	0.0357	0.0321	0.0397
R-FaIE ($\simeq$PLST)		0.0008	0.0008	0.0008	0.0008	0.0008	0.0455	0.0455	0.0453	0.0455	0.0453
P-LinearFaIE ($\sim$CPLST)		0.0015	0.0018	0.0020	0.0021	0.0021	0.0394	0.0491	0.0554	0.0581	0.0581
OP-FaIE		0.0019	0.0024	0.0024	0.0024	0.0026	0.0593	0.0596	0.0593	0.0591	0.0595
LinearFaIE		0.0021	0.0022	0.0023	0.0024	0.0024	0.0556	0.0638	0.0639	0.0659	0.0670
FaIE		**0.0023**	**0.0028**	**0.0028**	**0.0029**	**0.0028**	**0.0641**	**0.0640**	**0.0666**	**0.0669**	**0.0690**
kernel-CPLST		0.0009	0.0009	0.0009	0.0009	0.0009	0.0488	0.0483	0.0488	0.0483	0.0488
kernel-FaIE		**0.0038**	**0.0045**	**0.0054**	**0.0058**	**0.0062**	**0.0831**	**0.0834**	**0.0837**	**0.0840**	**0.0841**

Table 6. Average training costs (in seconds) of algorithms ("performing LSDR + training predictive models") with $L/K = 10\%$.

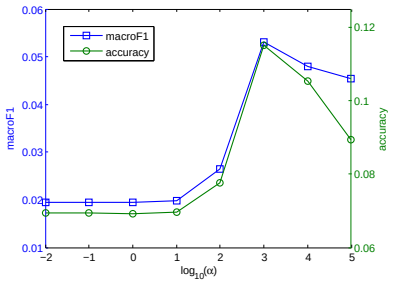
		delicious		*ESPGame*	
BR	L-SVM	28.168	$(0.000 + 28.168)$	377.035	$(0.000 + 377.035)$
	L-RR	49.993	$(0.000 + 49.993)$	101.940	$(0.000 + 101.940)$
CS		**5.057**	$(\mathbf{0.064 + 4.993})$	**10.364**	$(\mathbf{0.203 + 10.161})$
PLST		5.387	$(0.396 + 4.991)$	12.783	$(2.581 + 10.201)$
CPLST		7.080	$(2.132 + 4.948)$	16.684	$(6.486 + 10.198)$
MLC-BMaD		21.267	$(16.276 + 4.990)$	77.251	$(67.058 + 10.193)$
ML-CSSP		5.346	$(0.371 + 4.975)$	12.819	$(2.640 + 10.180)$
FaIE		10.012	$(5.032 + 4.980)$	20.110	$(9.909 + 10.201)$

all datasets, which demonstrates its effectiveness to handle non-linear correlations between the feature space and the latent space. Moreover, *kernel*-FaIE yields superior performance to *kernel*-CPLST, which outperforms CPLST. 8) On all datasets, as L/K increases, the performance of FaIE varies a little, which is due to the orthonormality constraint in formula (5) for enabling FaIE to compactly encode the original label space with a smaller L. Similar observations can be obtained on other variants, PLST and CPLST, which are also orthogonally constrained. Actually, we find that even with $L/K > 50\%$, this phenomenon still exists.

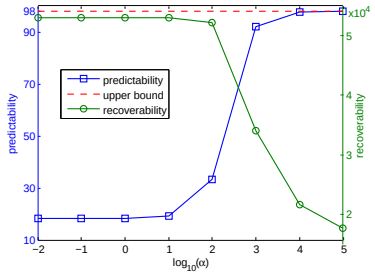
Moreover, in Table 6 we also report the average time costs for LSDR methods to perform LSDR and train predictive models over 5 runs on the largest *delicious* and *ESPGame*, with $L/K = 10\%$. As a reference, the time costs of BR are also reported. All algorithms are conducted with Matlab on a server with an Intel Xeon E5620 CPU and 24G RAM, except that BR with L-SVM is conducted using LIBLINEAR (Fan et al., 2008). Then we can draw the following observations. 1) Compared with BR, all LSDR methods can help to reduce the total training costs. 2) For performing LSDR, FaIE generally needs higher costs than CS, PLST, CPLST and ML-CSSP, though with superior classification performance. And it needs lower costs than MLC-BMaD, which performs boolean matrix decomposition for implicit encoding with a computational complexity of $\mathcal{O}(LNK^2)$.

4.3. Parameter analysis

Furthermore, we conduct experiments to see the effects of the only parameter α in the proposed FaIE. Fig. 2 gives an illustration of the variances of multi-label classification performance (sub-figure 2(a)) and the value fluctuations of predictability and recoverability in the objective function (sub-figure 2(b)) as α varies in $\{10^{-2}, 10^{-1}, \cdots, 10^{4}, 10^{5}\}$ in a run on *delicious* with $L/K = 10\%$. From the illustration we can draw the following observations. 1) The performance of FaIE, in terms of *macroF1* and *accuracy*, firstly increases and then decreases as α varies from 10^{-2} to 10^{5}. It further validates the reasonableness of jointly considering recoverability and predictability, as a good trade-off between both can yield a superior performance. 2) As α increases, the value of recoverability will decrease while



(a) Label-based macroF1 and example-based accuracy



(b) Values of recoverability and predictability

Figure 2. Effects of α in FaIE on the performance of multi-label classification (sub-figure 2(a)) and the values of recoverability and predictability (sub-figure 2(b)) on *delicious*, with $L/K = 10\%$ and the theoretical upper bound $\min(L, rank(\Delta)) = 98$.

that of predictability will increase and converge to its theoretical upper bound, *i.e.* $\min(L, rank(\Delta))$, as ensured by Lemma 1. Similar results are also obtained on other datasets and given in the supplementary appendix.

5. Conclusion

For tackling a multi-label classification problem with many classes, in this paper we propose an effective method termed FaIE to perform LSDR via feature-aware implicit label space encoding. Unlike most previous work, FaIE makes no assumptions concerning the encoding process. And it directly learns a feature-aware code matrix and a linear decoding matrix via jointly maximizing recoverability of the original label space and predictability of the low-dimensional latent space. Moreover, FaIE can also be specified to learn an explicit encoding function as previous work, and extended with kernel tricks to handle non-linear correlations between the feature space and the latent space.

Acknowledgments. This research was supported by the National Basic Research Project of China (Grant No. 2011CB70700), the National Natural Science Foundation of China (Grant No. 61271394, 61005045), and the National HeGaoJi Key Project (No. 2013ZX01039-002-002). The authors would like to thank anonymous reviewers and Prof. Tong Zhang for helpful comments and suggestions.

References

Argyriou, A., Evgeniou, T., and Pontil, M. Convex multi-task feature learning. *Mach. Learn.*, 73(3):243–272, 2008.

Balasubramanian, K. and Lebanon, G. The landmark selection method for multiple output prediction. In *ICML*, 2012.

Bi, W. and Kwok, J. Efficient multi-label classification with many labels. In *ICML*, 2013.

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

Carneiro, G., Chan, A.B., Moreno, P.J., and Vasconcelos, N. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):394–410, 2007.

Chen, Y. and Lin, H. Feature-aware label space dimension reduction for multi-label classification. In *NIPS*, 2012.

Dekel, O. and Shamir, O. Multiclass-multilabel classification with more classes than examples. In *AISTAT*, 2010.

Evgeniou, T., Micchelli, C.A., and Pontil, M. Learning multiple tasks with kernel methods. In *J. Mach. Learn. Res.*, pp. 615–637, 2005.

Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

Fürnkranz, J, Hüllermeier, E, Loza Mencía, E., and Brinker, K. Multilabel classification via calibrated label ranking. *Mach. Learn.*, 73(2):133–153, 2008.

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

Hariharan, B., Zelnik-Manor, L., Vishwanathan, S., and Varma, M. Large scale max-margin multi-label classification with priors. In *ICML*, 2010.

Hsu, D., Kakade, S.M., Langford, J., and Zhang, T. Multi-label prediction via compressed sensing. *NIPS*, 2009.

Jolliffe, I.T. *Principal component analysis*, volume 487. Springer-Verlag New York, 1986.

Kapoor, A., Viswanathan, R., and Jain, P. Multilabel classification using bayesian compressed sensing. In *NIPS*, 2012.

Katakis, I., Tsoumakas, G., and Vlahavas, I. Multilabel text classification for automated tag suggestion. In *ECML/PKDD*, 2008.

Lehoucq, R. B. and Sorensen, D. C. Deflation techniques for an implicitly restarted arnoldi iteration. *SIAM J. Matrix Anal. Appl.*, 17(4):789–821, 1996.

Lux, M. and Chatzichristofis, S. A. Lire: lucene image retrieval: an extensible java cbir library. In *MM*, 2008.

Needell, D. and Tropp, J.A. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Anal.*, 26(3):301–321, 2009.

Tai, F. and Lin, H. Multi-label classification with principle label space transformation. In *MLD*, 2010.

Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y., and Singer, Y. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6 (2):1453, 2006.

Tsoumakas, G. and Katakis, I. Multi-label classification: An overview. *Int. J. Data Warehous. Min.*, 3(3):1–13, 2007.

Tsoumakas, G., Katakis, I., and Vlahavas, I. Effective and efficient multilabel classification in domains with large number of labels. In *ECML/PKDD*, 2008a.

Tsoumakas, G., Katakis, I., and Vlahavas, I. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pp. 667–685. Springer, 2010.

Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., and Vlahavas, I. Mulan: A java library for multi-label learning. *J. Mach. Learn. Res.*, 12:2411–2414, 2011.

Tsoumakas, K.T.G., Kalliris, G., and Vlahavas, I. Multi-label classification of music into emotions. In *ICMIR*, 2008b.

Wicker, J., Pfahringer, B., and Kramer, S. Multi-label classification using boolean matrix decomposition. In *SAC*, 2012.

Zhang, M. and Zhou, Z. A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.*, PP(99):1–1, 2013.

Zhang, Y. and Schneider, J.G. Multi-label output codes using canonical correlation analysis. In *AISTAT*, 2011.

Zhou, T. and Tao, D. Multi-label subspace ensemble. In *AISTAT*, 2012.

Zhou, T., Tao, D., and Wu, X. Compressed labeling on distilled labelsets for multi-label learning. *Mach. Learn.*, 88(1-2):69–126, 2012.