Noname manuscript No.
(will be inserted by the editor)

Supervised Representation Learning for Multi-label Classification

Ming Huang, Fuzhen Zhuang[*], Xiao Zhang, Xiang Ao, Zhengyu Niu, Min-Ling Zhang, Qing He

Received: date / Accepted: date

Abstract Representation learning is one of the most important aspects of multi-label learning because of the intricate nature of multi-label data. Current researches of representation learning either fail to consider label knowledge or are weakened by the lack of labeled data. Moreover, most of them learn the representations and incorporate the label information in a two-step manner. In this paper, due to the success of representation learning by deep learning we propose a novel neural networks based framework named SERL to learn global feature representation by jointly considering all labels in an effective supervised manner. At its core, a two-encoding-layer autoencoder, which can utilize labeled and unlabeled data, is adopted to learn feature representation in the supervision of softmax regression. Specifically, the softmax regression incorporates label knowledge to improve the performance of both representation learning and multi-label learning by being jointly optimized with the autoencoder. Moreover, the autoencoder is expanded into two encoding layers to share knowledge with the softmax regression by sharing the second encoding weight matrix. We conduct extensive experiments on five real-world datasets to demonstrate the superiority of SERL over other state-of-the-art multi-label learning approaches.

keywords: Representation Learning, Multi-label Learning, Two-encoding-layer Autoencoder.

[*]Corresponding Author: zhuangfuzhen@ict.ac.cn, (8610)-62600765.
Ming Huang, Fuzhen Zhuang, Xiang Ao and Qing He are with the Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China. University of Chinese Academy of Sciences, Beijing 100049, China. Xiao Zhang is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China. Zhengyu Niu is with Baidu Inc., and Min-Ling Zhang is with School of Computer Science and Engineering, Southeast University, China.

1 Introduction

Multi-label learning, which deals with the problem where one object may be associated with one or more labels, has attracted extensive researches in the past decades [20]. Different from single-label problem where binary class and multi-class classification hold, multi-label learning could model the world more exactly. Besides, multi-label learning has widespread applications such as news classification and image processing [1]. For example, one news may belong to multiple topics such as politics and economy because it reports new policies on bank rate. A scenery picture, as a more familiar example, may contain sky, road, cornfield and so on, where they can be viewed as with multiple labels.

Traditionally in multi-label learning, the problem transformation method transformed the multi-label dataset to a series of single label datasets [31], such as the binary relevance method [21] and the label powerset method [22]. This kind of methods neglect the fact that some labels are more likely to co-exist in one instance, which is the main focus of many recent multi-label works. Therefore, in order to parameterize the label correlations, Ghamrawi et al. [7] proposed a multi-label classifier in conditional random field by modeling the label co-occurrences explicitly. Zhang et al. [28] utilized a bayesian network structure to encode the conditional dependencies of the labels and the feature set. Nguyen et al. [13] proposed a bayesian nonparametric approach to automatically learn the number of label-feature correlation patterns. However, most existing multi-label methods utilized the raw instance data to formalize the model, which might contain non-helpful feature attributes from the input space prior to training. Hence, learning better feature representation is important for the multi-label learning.

There exist some related works on multi-label learning classifiers based on representative features [33,15,27]. MMDM [33] discovers a low-dimension feature space which maximizes the dependence between the original features and the corresponding labels. LIFT [27] uses clustering techniques to construct label-specific features for each label and then solves binary classification problems based on the transformed features. MLFE [32] utilizes the structural information in feature space to enrich the labeling information. However, these works either learn representative features without considering label knowledge or suffer from the lack of labeled data. Recently, deep learning has proven to be able to learn good representation in natural language processing, image classification, and so on. And some effort has been devoted to handling multi-label learning problem to improve the performance. Read et al. [15] used restricted boltzmann machine (RBM) to get a better representation of the original features, and then applied the supervised learning algorithms to training classification models. However, they performed the optimization framework in a two-step manner, while we try to learn the representation and incorporate label knowledge in a joint optimization framework.

To address these issues, we propose a novel framework named SERL (*S*up*E*rvised *R*epresentation *L*earning for multi-label classification) in this paper. SERL adopts a two-encoding-layer autoencoder to learn better representation of the original features in the supervision of softmax regression. Specially, the softmax regression incorporates label knowledge to improve the performance of both representation learning and multi-label learning by being jointly optimized with the autoencoder, where the autoencoder can sufficiently utilize labeled and unlabeled data to learn nonlinear representation of the original features. In addition, the autoencoder is expanded into two encoding layers to share knowledge with the softmax regression by sharing the second encoding weight matrix. We evaluate the proposed approach on five real-world datasets and observe the effectiveness of SERL that it can outperform the compared state-of-the-art algorithms significantly. The contribution of this paper is summarized as follows.

- We propose an autoencoder based framework (SERL) to discover latent knowledge of the original features by jointly considering all labels in an effective supervised manner.
- The autoencoder learns representation from labeled and unlabeled data in the supervision of the softmax regression. Moreover, the autoencoder shares knowledge with softmax regression by sharing the second encoding weight matrix.
- We conduct extensive experiments on five real-world datasets to demonstrate the superiority of the proposed method over other state-of-the-art algorithms.

The remainder of this paper is organized as follows. Section 2 introduces the preliminary knowledge. The framework and its solution are detailed in Section 3. The experimental results are reported in Section 4. Section 5 discusses the related work and finally Section 6 concludes.

2 Preliminary Knowledge

2.1 Softmax Regression

Softmax regression which is often used to solve the problem of multi-class classification can be regarded as the generalization of the logistic regression. When given a test input x, softmax regression estimates the probability of each label (label space $y \in \{1, 2, ..., k\}$) by the hypothesis function as follows,

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \begin{bmatrix} p(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\theta}) \\ p(y_i = 2|\boldsymbol{x}_i; \boldsymbol{\theta}) \\ \vdots \\ p(y_i = k|\boldsymbol{x}_i; \boldsymbol{\theta}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}}} \begin{bmatrix} e^{\boldsymbol{\theta}_1^\top \boldsymbol{x}_i} \\ e^{\boldsymbol{\theta}_2^\top \boldsymbol{x}_i} \\ \vdots \\ e^{\boldsymbol{\theta}_k^\top \boldsymbol{x}_i} \end{bmatrix}. \tag{1}$$

The objective function of softmax regression can be described as follows,

$$\min_{\boldsymbol{\theta}} \left(-\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} 1\{y_i = j\} \log \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}_i}}{\sum_{l=1}^{k} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}} \right), \tag{2}$$

where the indicator function $1\{\cdot\}$ equals 1 when x_i holds label j and equals 0 otherwise. Given training dataset $\{\boldsymbol{x_i},\ \boldsymbol{y_i}\}_{i=1}^{n}$ ($y_i \in \{1, 2, ..., k\}$), the model parameter

$\boldsymbol{\theta}$ can be derived by minimizing Eq. (2). After training, the probability of each label can be computed using Eq. (1), then the predicted label can be assigned as follows,

$$y = \max_{j} \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}}}{\sum_{l=1}^{k} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}}}. \tag{3}$$

2.2 Autoencoder

Autoencoder, which is a neural network, uses unsupervised learning method to learn compressed features from original features. A multi-layer autoencoder comprises one input layer, one output layer, and several hidden layers. The aim of the autoencoder is to reconstruct the input signal in the output layer with the least amount of distortion. A simple autoencoder consists of two parts, that is, an encoder including the input layer and hidden layer and a decoder including the hidden layer and output layer. Given an input $\boldsymbol{x_i} \in \mathbb{R}^{d\times 1}$, weight matrix $\boldsymbol{W}_1 \in \mathbb{R}^{k\times d}$, $\boldsymbol{W}_1^{'} \in \mathbb{R}^{d\times k}$, and bias vector $\boldsymbol{b}_1 \in \mathbb{R}^{k\times 1}$, $\boldsymbol{b}_1^{'} \in \mathbb{R}^{d\times 1}$, a single hidden layer autoencoder encodes it into the hidden layer $\boldsymbol{\xi_i} \in \mathbb{R}^{k\times 1}$ and decodes the hidden layer into the output layer $\hat{\boldsymbol{x_i}}$ which is as same as possible with the input layer. This process can be described as,

$$\boldsymbol{\xi}_i = f(\boldsymbol{W}_1\boldsymbol{x}_i + \boldsymbol{b}_1), \quad \hat{\boldsymbol{x}}_i = f(\boldsymbol{W}_1^{'}\boldsymbol{\xi}_i + \boldsymbol{b}_1^{'}). \tag{4}$$

Here we use the sigmoid function as the activation function f. Given a set of inputs $\{\boldsymbol{x}_i\}_{i=1}^{n}$, the goal of autoencoder is to minimize the reconstruction error using L2 regularization as follows,

$$\min_{\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_1', \boldsymbol{b}_1'} \sum_{i=1}^{n} \|\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i\|^2. \tag{5}$$

3 The SERL Framework

In this section, we present our proposed framework in detail and the symbols used are listed in Table 1.

3.1 Problem Formalization

The proposed framework is composed of a two-encoding-layer autoencoder and softmax regression as shown in Fig 1. The two components are jointly optimized and they share the second encoding weight matrix $\boldsymbol{W}_2$. Given multi-label training dataset $D_r = \{(x_i^{(r)}, Y_i^{(r)}) | 1 \le i \le n_r\}$ and test dataset $D_s = \{(x_i^{(s)}, Y_i^{(s)}) | 1 \le i \le n_s\}$, where $x_i^{(r)}, x_i^{(s)} \in \mathbb{R}^{d\times 1}$ and $Y_i^{(r)}, Y_i^{(s)} \subseteq \mathcal{Y}$ ($\mathcal{Y} = \{1, 2.., c\}$) are sets of relevant labels associated with $x_i^{(r)}, x_i^{(s)}$ respectively. The objective function can be described as follows,

$$\mathcal{J} = \sum_{t\in\{r,s\}} J(\boldsymbol{x}^{(t)}, \hat{\boldsymbol{x}}^{(t)}) + \alpha\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\xi}^{(r)}) + \beta\Omega(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{W}^{'}, \boldsymbol{b}'). \tag{6}$$

Table 1: Notations and Denotations

$\mathcal{D}_r, \mathcal{D}_s$	The training and test dataset
n_r	The number of instances in training dataset
n_s	The number of instances in test dataset
d	The number of nodes in the input layer
k	The number of nodes in the embedding layer
c	The number of nodes in the label layer, also the number of labels
$\boldsymbol{x}_i^{(r)}, \boldsymbol{x}_i^{(s)}$	The i-th instance of training and test dataset
$\hat{\boldsymbol{x}}_i^{(r)}, \hat{\boldsymbol{x}}_i^{(s)}$	The reconstructions of $\boldsymbol{x}_i^{(r)}$ and $\boldsymbol{x}_i^{(s)}$
$Y_i^{(r)}, Y_i^{(s)}$	The label sets of instance $\boldsymbol{x}_i^{(r)}$ and $\boldsymbol{x}_i^{(s)}$
$\boldsymbol{\xi}_i^{(r)}, \boldsymbol{\xi}_i^{(s)}$	The hidden representations of $\boldsymbol{x}_i^{(r)}$ and $\boldsymbol{x}_i^{(s)}$
$\hat{\boldsymbol{\xi}}_i^{(r)}, \hat{\boldsymbol{\xi}}_i^{(s)}$	The reconstructions of $\boldsymbol{\xi}_i^{(r)}$ and $\boldsymbol{\xi}_i^{(s)}$
$\boldsymbol{z}_i^{(r)}, \boldsymbol{z}_i^{(s)}$	The hidden representations of $\boldsymbol{\xi}_i^{(r)}$ and $\boldsymbol{\xi}_i^{(s)}$
$\boldsymbol{W}_i, \boldsymbol{b}_i$	Encoding weight and bias matrix for layer i
$\boldsymbol{W}_i^{'}, \boldsymbol{b}_i^{'}$	Decoding weight and bias matrix for layer i
$\top$	The transposition of a matrix
$\circ$	The element-wise product of vectors or matrixes

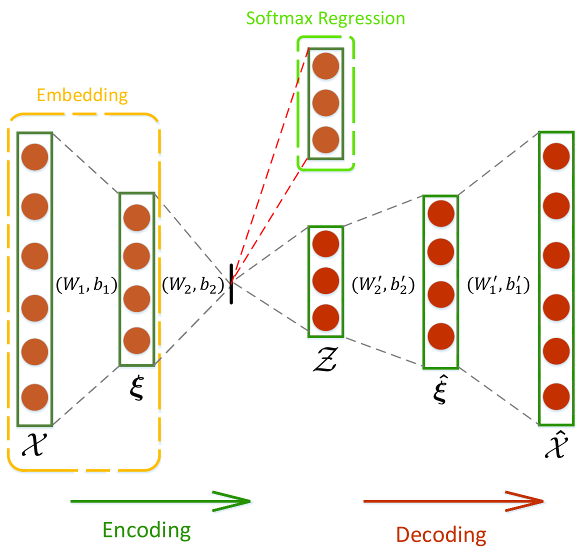

Fig. 1: The Framework of SERL

where J is the loss of autoencoder, $\mathcal{L}$ is the loss of softmax regression, Ω is the regularization term, α and β are trade-off parameters for the whole framework. $\boldsymbol{W}, \boldsymbol{b}$ include all the parameters for encoding, and $\boldsymbol{W}^{'}, \boldsymbol{b}'$ represent the ones for decoding.

There are three terms in Eq. (6). In the first term $J(\boldsymbol{x}^{(t)}, \hat{\boldsymbol{x}}^{(t)})$, the reconstruction error is calculated for both training and test datasets, and it is defined as follows,

$$J(\boldsymbol{x}^{(t)}, \hat{\boldsymbol{x}}^{(t)}) = \sum_{t\in\{r,s\}} \sum_{i=1}^{n_t} ||\boldsymbol{x}_i^{(t)} - \hat{\boldsymbol{x}}_i^{(t)}||^2, \tag{7}$$

where

$$\boldsymbol{\xi}_i^{(t)} = f(\boldsymbol{W}_1 \boldsymbol{x}_i^{(t)} + \boldsymbol{b}_1),\ \boldsymbol{z}_i^{(t)} = f(\boldsymbol{W}_2 \boldsymbol{\xi}_i^{(t)} + \boldsymbol{b}_2), \tag{8}$$

$$\hat{\boldsymbol{\xi}}_i^{(t)} = f(\boldsymbol{W}_2^{'} \boldsymbol{z}_i^{(t)} + \boldsymbol{b}_2^{'}),\ \hat{\boldsymbol{x}}_i^{(t)} = f(\boldsymbol{W}_1^{'} \hat{\boldsymbol{\xi}}_i^{(t)} + \boldsymbol{b}_1^{'}). \tag{9}$$

There are three hidden layers in our framework. The first one called the embedding layer has k nodes ($k \leq d$) with output $\boldsymbol{\xi}_i^{(t)} \in \mathbb{R}^{k\times 1}$, weight matrix $\boldsymbol{W}_1 \in \mathbb{R}^{k\times d}$, and bias vector $\boldsymbol{b}_1 \in \mathbb{R}^{k\times 1}$. The second one called the label layer has c nodes (equals to the number of labels) with output $\boldsymbol{z}_i^{(t)} \in \mathbb{R}^{c\times 1}$, weight matrix $\boldsymbol{W}_2 \in \mathbb{R}^{c\times k}$ and bias vector $\boldsymbol{b}_2 \in \mathbb{R}^{c\times 1}$. The input of the label layer is also the input of the softmax regression which incorporates label knowledge. The third one is the reconstruction of the embedding layer with output $\hat{\boldsymbol{\xi}}_i^{(t)}$, weight matrix $\boldsymbol{W}_2^{'} \in \mathbb{R}^{k\times c}$ and bias vector $\boldsymbol{b}_2^{'} \in \mathbb{R}^{k\times 1}$. The output layer is the reconstruction of input $\boldsymbol{x}_i^{(t)}$ with output $\hat{\boldsymbol{x}}_i^{(t)} \in \mathbb{R}^{d\times 1}$, weight matrix $\boldsymbol{W}_1^{'} \in \mathbb{R}^{d\times k}$ and bias vector $\boldsymbol{b}_1^{'} \in \mathbb{R}^{d\times 1}$.

The second term in the objective Eq. (6) is the optimization of softmax regression, which incorporates the label knowledge from training data. Note here that the autoencoder is expanded into two encoding layers to share the second encoding weight matrix W_2 with the softmax regression, which aims to share knowledge with the softmax regression.

Here we try to use the softmax regression to handle multi-label data. The basic idea is to transform the multi-label data to multi-class data. Let $\sigma : (x_i, Y_i) \rightarrow \{(x_i, y_j)|y_j \in Y_i\}$ be the function which converts a (instance, labels) pair into a set of (instance, label) pair where each (instance, label) pair contains only one label. For example, suppose we have one instance x_i with labels y_1, y_2, y_4. σ converts $(x_1, \{y_1, y_2, y_4\})$ to (x_1, y_1), (x_1, y_2), (x_1, y_4). In the training phase, we firstly converts the original multi-label training dataset D_r into the following multi-class training dataset $D_r^{\dagger}$ by σ as follows,

$$D_r^{\dagger} = \{\sigma(x_i, Y_i)|1 \leq i \leq n_r\}. \tag{10}$$

After that, softmax regression $\mathcal{M}$ is utilized to induce multi-class classifier $g^{\dagger} : \mathcal{X} \rightarrow \mathcal{Y}, i.e., g^{\dagger} \leftarrow \mathcal{M}(D_r^{\dagger})$ ($\mathcal{X} \in \mathbb{R}^{d\times 1}$, $\mathcal{Y} = \{1, 2, ..., c\}$). The objective function of softmax regression can be formalized as follows,

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\xi}^{(r)}) = -\frac{1}{n_r} \sum_{i=1}^{n_r} \sum_{j=1}^{c} 1\{y_i^{(r)} = j\} \log \frac{e^{\boldsymbol{\theta}_j^{\top} \boldsymbol{\xi}_i^{(r)}}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^{\top} \boldsymbol{\xi}_i^{(r)}}}.$$

In this term, $\boldsymbol{\xi}_i^{(r)}$ is the output of the embedding layer and $\boldsymbol{\theta}_j^{\top}$ ($j \in \{1, ..., c\}$) is the j-th row of $\boldsymbol{W}_2$ which is also the second encoding weight matrix of autoencoder.

Finally, the last term in the objective Eq. (6) is the regularization on model parameters which controls the complexity of the framework to improve its generalization ability. The last term is defined as follows,

$$\begin{aligned}\Omega(\boldsymbol{W},\boldsymbol{b},\boldsymbol{W}^{'},\boldsymbol{b}') = \|\boldsymbol{W}_1\|^2 + \|\boldsymbol{b}_1\|^2 + \|\boldsymbol{W}_2\|^2 + \\ \|\boldsymbol{b}_2\|^2 + \|\boldsymbol{W}_1^{'}\|^2 + \|\boldsymbol{b}_1^{'}\|^2 + \|\boldsymbol{W}_2^{'}\|^2 + \|\boldsymbol{b}_2^{'}\|^2.\end{aligned} \tag{11}$$

3.2 Solution of the Proposed Framework

The optimization problem of our proposed framework is to minimize $\mathcal{J}$ (seen in Eq. (6)) as a function of $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_2$, $\boldsymbol{W}_2^{'}$, $\boldsymbol{b}_2^{'}$, $\boldsymbol{W}_1^{'}$ and $\boldsymbol{b}_1^{'}$. This is an unconstrained optimization problem and therefore we can adopt the gradient descent method to solve it.

We first introduce some intermediate variables for simplicity as follows,

$$\begin{aligned} A_i^{(t)} &= \left(\hat{\boldsymbol{x}}_i^{(t)} - \boldsymbol{x}_i^{(t)}\right) \circ \hat{\boldsymbol{x}}_i^{(t)} \circ \left(1 - \hat{\boldsymbol{x}}_i^{(t)}\right), \\ B_i^{(t)} &= \hat{\boldsymbol{\xi}}_i^{(t)} \circ \left(1 - \hat{\boldsymbol{\xi}}_i^{(t)}\right), \\ C_i^{(t)} &= \boldsymbol{z}_i^{(t)} \circ \left(1 - \boldsymbol{z}_i^{(t)}\right), \quad D_i^{(t)} = \boldsymbol{\xi}_i^{(t)} \circ \left(1 - \boldsymbol{\xi}_i^{(t)}\right). \end{aligned} \tag{12}$$

The partial derivatives of $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{W}_2^{'}$, $\boldsymbol{W}_1^{'}$ are as follows respectively,

$$\begin{aligned} \frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1} = &\sum_{t\in\{r,s\}} \sum_{i=1}^{n_t} 2\boldsymbol{W}_2^{\top} (\boldsymbol{W}_2^{'\top} (\boldsymbol{W}_1^{'\top} A_i^{(t)} \circ B_i^{(t)}) \circ C_i^{(t)}) \circ D_i^{(t)} \boldsymbol{x}_i^{(t)\top} \\ &- \frac{\alpha}{n_r} \sum_{i=1}^{n_r} \sum_{j=1}^{c} 1\{y_i^{(r)} = j\} (\boldsymbol{W}_{2j}^{\top} - \frac{\boldsymbol{W}_2^{\top} e^{\boldsymbol{W}_2 \boldsymbol{\xi}_i^{(r)}}}{\sum_l e^{\boldsymbol{W}_{2l} \boldsymbol{\xi}_i^{(r)}}}) \circ D_i^{(r)} \boldsymbol{x}_i^{(r)\top} \\ &+ 2\beta \boldsymbol{W}_1, \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_{2j}} = &\sum_{t\in\{r,s\}} \sum_{i=1}^{n_t} 2\boldsymbol{W}_{2j}^{'\top} (\boldsymbol{W}_1^{'\top} A_i^{(t)} \circ B_i^{(t)}) \circ C_{ij}^{(t)} \boldsymbol{\xi}_i^{(t)\top} \\ &- \frac{\alpha}{n_{rj}} (\sum_{i=1}^{n_{rj}} \boldsymbol{\xi}_i^{(r)\top} - \sum_{i=1}^{n_r} \frac{e^{\boldsymbol{W}_{2j} \boldsymbol{\xi}_i^{(r)}}}{\sum_l e^{\boldsymbol{W}_{2l} \boldsymbol{\xi}_i^{(r)}}} \boldsymbol{\xi}_i^{(r)\top}) + 2\beta \boldsymbol{W}_{2j}, \end{aligned} \tag{13}$$

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_2^{'}} = \sum_{t\in\{r,s\}} \sum_{i=1}^{n_t} 2\boldsymbol{W}_1^{'\top} A_i^{(t)} \circ B_i^{(t)} \boldsymbol{z}_i^{(t)\top} + 2\beta \boldsymbol{W}_2^{'}, \tag{14}$$

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1^{'}} = \sum_{t\in\{r,s\}} \sum_{i=1}^{n_t} 2A_i^{(t)} \hat{\boldsymbol{\xi}}_i^{(t)\top} + 2\beta \boldsymbol{W}_1^{'}, \tag{15}$$

where $\boldsymbol{W}_{2j}$ is the j-th row of $\boldsymbol{W}_2$ and n_{rj} is the number of instance associated with label j in training dataset. According to the above partial derivatives, we update the parameters by alternatively iterating following those rules,

$$\begin{aligned}
&\boldsymbol{W}_1 \leftarrow \boldsymbol{W}_1 - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{W}_1}, \quad \boldsymbol{b}_1 \leftarrow \boldsymbol{b}_1 - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{b}_1},\\
&\boldsymbol{W}_1^{'} \leftarrow \boldsymbol{W}_1^{'} - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{W}_1^{'}}, \quad \boldsymbol{b}_1^{'} \leftarrow \boldsymbol{b}_1^{'} - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{b}_1^{'}},\\
&\boldsymbol{W}_2 \leftarrow \boldsymbol{W}_2 - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{W}_2}, \quad \boldsymbol{b}_2 \leftarrow \boldsymbol{b}_2 - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{b}_2},\\
&\boldsymbol{W}_2^{'} \leftarrow \boldsymbol{W}_2^{'} - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{W}_2^{'}}, \quad \boldsymbol{b}_2^{'} \leftarrow \boldsymbol{b}_2^{'} - \eta\frac{\partial\mathcal{J}}{\partial\boldsymbol{b}_2^{'}}.
\end{aligned} \tag{16}$$

where η is step length controlling the learning rate. Finally, the whole algorithm is summarized in Algorithm 1.

Algorithm 1: *S*up*E*rvised *R*epresentation *L*earning for multi-label classification (SERL)

```
1 Input: Training dataset $D_r = \{(x_i^{(r)}, Y_i^{(r)})|1 \le i \le n_r\}$ and test dataset
     $D_s = \{(x_i^{(s)}, Y_i^{(s)})|1 \le i \le n_s\}$, the number of nodes in the embedding layer $k$ and label
     layer $c$, trade-off parameters $\alpha$, $\beta$.
2 Output: The predicted label set $Y_i^{(s)}$ of each test instance $x_i^{(s)}$ .
    1. Convert training dataset according to Eq. (10);
    2. Initialize $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_2^{'}, \boldsymbol{W}_1^{'}$ and $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_2^{'}, \boldsymbol{b}_1^{'}$ by SDAE which is trained on both training and test
       dataset;
    3. Compute the partial derivatives of all variables based on Eqs. (13), (13) (14) and (15);
    4. Update the variables iteratively using Eq. (16);
    5. Continue Step3 and Step4 until the algorithm converges;
    6. Predict the label sets of test instances.
```

Although the optimization of the objective function is not convex, we can get a better local optimal solution through appropriate initialization of the weights and biases. Specifically, we use the Stacked Denosing AutoEncoder (SDAE) to initialize the values of $\boldsymbol{W}$ and $\boldsymbol{b}$.

3.3 Prediction

After training, we use the softmax regression to predict the label set of each test instance. Specifically, we can estimate the probability $P(y_i^{(s)} = j|\boldsymbol{x}_i^{(s)})$ of one certain test instance belonging to each label. Then we sort all the label probabilities in descending order and compute the difference between two adjacent label probabilities in this order. Finally we assign the labels, which are in the front of the position of the max difference, as the predicted labels of the instance. This process can be described by Fig 2, where P_i is the probability of one certain test instance belonging to label i and $\triangle P_j$ is the probability difference between adjacent labels in the ordered list.

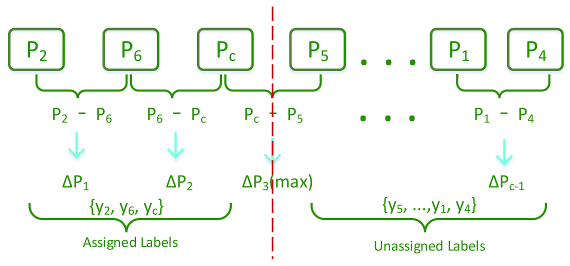

Fig. 2: The Prediction Strategy

4 Experimental Evaluation

In this section, we conduct extensive experiments on five benchmark multi-label datasets to evaluate the performance of the proposed framework.

4.1 Datasets and Preprocessing

These five datasets include slashdot, corel5k, bibtex, corel16k01 (sample 1) and corel16k02 (sample 2) from MULAN [23] and MEKA [17] multi-label learning libraries. These datasets can evaluate the proposed framework in different cases including text and image. For all the datasets, we randomly sample 50% of examples without replacement to construct training dataset and the remaining 50% to construct test dataset. We sample each dataset for five times and calculate the average accuracies. The information of all the datasets is detailed in Table 2, #instances represents the number of instances, #features represents the feature dimension, #labels represents the number of labels, and #domains represents the domains of the datasets.

Table 2: Datasets Infomration

Datasets	#instances	#features	#labels	#domains
slashdot	3782	1079	22	text
corel5k	5000	499	374	image
bibtex	7395	1836	159	text
corel16k01	13766	500	153	image
corel16k02	13761	500	164	image

4.2 Comparison Methods

We compare our proposed model with seven multi-label algorithms as follows.

• ***Binary Relevance(BR)*** [1] This algorithm learns c independent binary classifiers for each label and queries all the classifiers for prediction.

• ***Calibrated Label Ranking(CLR)*** [6] This algorithm uses pairwise comparison to decompose the multi-label learning problem into the label ranking problem with calibrated scenario.

• ***Random k-Labelsets(RAkEL)*** [24] This algorithm applies Label Powerset techniques, which transforms the multi-label learning problem into the multi-class classification problems, on an ensemble of k random label subsets.

• ***Ensemble of Classifier Chains(ECC)*** [16] This algorithm is an ensemble of classifier chain algorithm which considers high-order relations among labels represented in an ordered chain and then trains c binary classifiers according to the chain.

• ***Multi-label learning with Label specIfic FeaTures (LIFT)*** [27] This algorithm conducts clustering analysis on positive and negative instances of each label to construct its specific features, then applies binary relevance algorithm on label-specific features of each label.

• ***Multi-label learning with Stacked Denoising AutoEncoders (SDAE)*** [25] Here we use SDAE in a two-step manner to compare with our joint optimization framework. Specially, we first train the autoencoder alone to learn feature representation and then combine the new features with the labels to construct new training dataset. Finally, we use Bayesian Multinomial Regression(BMR) to learn the classifier on the new training dataset.

• ***Multi-label Learning with Feature-induced labeling information Enrichment (MLFE)*** [32] In MLFE, the structural information in feature space is utilized to enrich the labeling information. The sparse reconstruction among the training examples is conducted to characterize the underlying structure of feature space. Then the reconstruction information is conveyed from feature space to label space so as to enrich the labeling information.

4.3 Experimental Settings

There are three factors in our proposed framework including trading-off parameters α, β and the number of nodes k of the embedding layer. After cross-validations on training dataset, we set $\alpha = 15$, $\beta = 0.005$, $k = 100$ for all datasets. LIBSVM with linear kernel [2] is employed as the base classifier for all baselines except SDAE. Bayesian Multinomial Regression(BMR) [12] is employed as the base classifier for SDAE. Specifically, for RAkEL, the size of label subset k is set as 3 and the size of ensemble is set as $2c$ (c is the number of labels) as a rule-of-thumb setting. For ECC, the size of ensemble is set 100 to cover the high-order relations among labels sufficiently. For LIFT, the ratio is set to 0.1 as reported in their original paper [27]. For MLFE, the penalty parameters β_1, β_2 and β_3 are set as 2, 10, 1, respectively according to [32].

4.4 Results and Discussion

To compare our proposed model with baselines in a more comprehensive way, we adopt two types of evaluation metrics, i.e., ranking metrics and classification metrics.

Table 3: Multi-label learning performance comparison on ranking evaluation metrics on five data sets

Datasets	Methods	OneError	Coverage	RankingLoss	AvgPrecision	MacroAUC
slashdot	BR	.4394±.0093	.1235±.0011	.1070±.0008	.6610±.0045	.8188±.0165
	CLR	.9944±.0007	.3029±.0060	.4484±.0038	.1973±.0050	.7263±.0156
	RAkEL	.4363±.0065	.1617±.0017	.1414±.0012	.6478±.0041	.7657±.0174
	ECC	.4122±.0074	.1128±.0015	.0966±.0016	.6842±.0044	**.8592±.0064**
	LIFT	.4307±.0022	.1172±.0023	.1017±.0024	.6694±.0020	.8512±.0061
	SDAE	.5863±.0083	.1606±.0008	.1451±.0010	.5527±.0038	.7786±.0046
	MLFE	**.3923±.0074**	.1295±.0036	.1105±.0035	**.6947±.0048**	.5962±.0041
	SERL	.4089±.0024	**.1073±.0033**	**.0921±.0031**	.6927±.0023	.8451±.0073
corel5k	BR	.6872±.0040	.3036±.0020	.1299±.0010	.2693±.0027	.5947±.0028
	CLR	.6595±.0145	.3054±.0125	.1431±.0025	.2520±.0195	.6413±.0074
	RAkEL	.6905±.0041	.4301±.0045	.1912±.0024	.2505±.0030	.5735±.0071
	ECC	.6855±.0054	.2901±.0027	.1240±.0009	.2845±.0029	.6491±.0077
	LIFT	.7007±.0131	.3154±.0040	.1317±.0011	.2829±.0049	.6799±.0043
	SDAE	.7471±.0026	.3230±.0019	.1423±.0011	.2395±.0016	.6208±.0029
	MLFE	.6752±.0116	.4635±.0008	.2090±.0030	.2700±.0041	.5078±.0003
	SERL	**.6445±.0059**	**.2542±.0032**	**.1063±.0014**	**.3152±.0023**	**.7120±.0052**
bibtex	BR	.4090±.0068	.1656±.0020	.0907±.0009	.5299±.0023	.8685±.0038
	CLR	.3797±.0104	.1136±.0019	.0644±.0011	.5640±.0042	.9095±.0014
	RAkEL	.4050±.0041	.2427±.0024	.1359±.0016	.5063±.0017	.8163±.0036
	ECC	.3807±.0053	.1389±.0020	.0741±.0011	.5702±.0015	.8977±.0026
	LIFT	.4069±.0036	.1495±.0034	.0824±.0021	.5439±.0013	.9040±.0025
	SDAE	.5862±.0042	.1945±.0022	.1228±.0017	.3841±.0041	.8484±.0022
	MLFE	**.3777±.0067**	.1695±.0035	.0889±.0018	.5733±.0043	.5381±.0010
	SERL	.3997±.0064	**.1061±.0026**	**.0579±.0015**	**.5740±.0044**	**.9276±.0016**
corel16k01	BR	.7248±.0070	.3202±.0022	.1643±.0014	.2818±.0029	.6523±.0041
	CLR	.6468±.0048	.2750±.0018	.1415±.0012	.3229±.0021	.7286±.0013
	RAkEL	.7235±.0064	.4097±.0037	.2128±.0022	.2647±.0023	.6076±.0026
	ECC	.6773±.0040	.3042±.0022	.1550±.0014	.3174±.0027	.6908±.0032
	LIFT	.6944±.0052	.3268±.0050	.1652±.0027	.3086±.0044	.6958±.0017
	SDAE	.7364±.0029	.3280±.0020	.1719±.0009	.2793±.0013	.6605±.0022
	MLFE	.6695±.0040	.3682±.0024	.1891±.0017	.3192±.0017	.5105±.0002
	SERL	**.6460±.0043**	**.2477±.0027**	**.1269±.0018**	**.3517±.0024**	**.7721±.0012**
corel16k02	BR	.7268±.0087	.3138±.0029	.1601±.0007	.2731±.0045	.6641±.0030
	CLR	**.6326±.0206**	.2647±.0029	.1359±.0007	.3149±.0032	.7418±.0029
	RAkEL	.7272±.0081	.3942±.0027	.2045±.0009	.2542±.0043	.6096±.0033
	ECC	.6720±.0078	.2932±.0026	.1485±.0006	.3145±.0017	.7024±.0035
	LIFT	.6833±.0043	.3165±.0041	.1596±.0022	.3073±.0022	.7089±.0047
	SDAE	.7301±.0064	.3207±.0024	.1673±.0009	.2756±.0019	.6696±.0036
	MLFE	.6645±.0042	.3663±.0014	.1876±.0004	.3171±.0017	.5111±.0006
	SERL	.6446±.0047	**.2408±.0022**	**.1229±.0008**	**.3458±.0019**	**.7794±.0010**

Further more, both types of metrics can be subdivided into example-based and label-based ones. Table 3 and Table 4 summarizes the results on all five datasets. Next, we analyze the results on all these metrics in detail as follows.

4.4.1 Results on Ranking Evaluation Metrics

Among all ranking evaluation metrics, OneError, Coverage, RankingLoss and Avg-Precision are example-based, while MacroAUC is label-based.

Table 4: Multi-label learning performance comparison on classification evaluation metrics on five data sets

Datasets	Methods	Accuracy	F1	MacroF1
slashdot	BR	.1219±.0018	.1843±.0026	.2502±.0084
	CLR	.0022±.0005	.0024±.0005	.1184±.0218
	RAkEL	.3506±.0085	.3628±.0088	.3388±.0112
	ECC	.4271±.0029	.4433±.0032	.3908±.0193
	LIFT	.3489±.0098	.3615±.0098	.3684±.0188
	SDAE	.3754±.0050	.4100±.0053	.3206±.0196
	MLFE	.3512±.0060	.3624±.0064	.3367±.0214
	SERL	**.5323±.0030**	**.5694±.0032**	**.4255±.0183**
corel5k	BR	.0029±.0003	.0054±.0005	.0680±.0053
	CLR	.1115±.0065	.1900±.0091	.0766±.0079
	RAkEL	.0137±.0005	.0191±.0004	.0686±.0058
	ECC	.0565±.0018	.0792±.0021	.0781±.0049
	LIFT	.0398±.0036	.0555±.0052	.0829±.0048
	SDAE	.0979±.0012	.1498±.0019	.0847±.0056
	MLFE	.0771±.0029	.1099±.0036	.0830±.0052
	SERL	**.1399±.0029**	**.2079±.0036**	**.0866±.0051**
bibtex	BR	.2742±.0019	.3235±.0027	.2187±.0052
	CLR	.2718±.0107	.3182±.0149	.2235±.0046
	RAkEL	.2742±.0019	.3231±.0028	.2178±.0047
	ECC	.2796±.0032	.3280±.0038	.2130±.0032
	LIFT	.2477±.0058	.2939±.0057	.1913±.0086
	SDAE	.2303±.0041	.2851±.0040	.1183±.0034
	MLFE	.2149±.0064	.2549±.0073	.1037±.0017
	SERL	**.3508±.0035**	**.4261±.0042**	**.2249±.0041**
corel16k01	BR	.0136±.0018	.0191±.0025	.0094±.0019
	CLR	.0133±.0018	.0185±.0025	.0076±.0015
	RAkEL	.0157±.0017	.0222±.0024	.0087±.0013
	ECC	.0523±.0011	.0713±.0013	.0258±.0014
	LIFT	.0218±.0043	.0300±.0058	.0215±.0020
	SDAE	.1239±.0008	.1817±.0013	.0443±.0024
	MLFE	.0728±.0024	.0998±.0032	.0375±.0007
	SERL	**.1640±.0023**	**.2321±.0029**	**.0573±.0012**
corel16k02	BR	.0162±.0017	.0234±.0024	.0139±.0017
	CLR	.0153±.0013	.0221±.0020	.0126±.0013
	RAkEL	.0157±.0018	.0227±.0026	.0136±.0014
	ECC	.0459±.0017	.0636±.0023	.0278±.0028
	LIFT	.0245±.0024	.0338±.0032	.0314±.0022
	SDAE	.1234±.0017	.1812±.0025	.0483±.0020
	MLFE	.0731±.0022	.1021±.0029	.0393±.0021
	SERL	**.1599±.0029**	**.2292±.0039**	**.0530±.0020**

– We can see that SERL performs the best in all five datasets on Coverage and RankingLoss. Even on the metric of OneError, SERL achieves the best performance on datasets corel5k and corel16k01, and gets an comparable performance to CLR and MLFE, which obtain best results on some corresponding datasets.
– For label-based ranking metric MacroAUC, SERL also achieves the best performance in most datasets. According to Table 2, we can see that corel5k has the most labels up to 374 and slashdot has the least labels of 22. The results in all the

five datasets show the outstanding performance of SERL in probability estimation over the datasets with high-diversity of label size.

4.4.2 Results on Classification Evaluation Metrics

Among classification evaluation metrics, Accuracy and F1 are example-based and MacroF1 is label-based.

– It is obvious that SERL achieves better performance than the baselines in terms of Accuracy and F1. We can get the following two observations from the results. The first one is that our model performs well for each example, which contributes to high accuracy and F1. And the other one is that some baselines such as BR and RAkEL output empty sets for some examples, failing to predict label information, which makes no sense for classification and leads to unsatisfying results.
– For MacroF1, SERL achieves the best in all data sets, which proves good performance of SERL in classifying the positive and negative examples of each label. The fact that SERL does well in both example-based and label-based classification metrics shows the superior classification performance of our model.

Overall, all the results validate the effectiveness of our framework.

4.5 Parameter Sensitivity

For analyzing the influence of the parameters α, β, k, we do a series of sensitivity experiments. We choose RankingLoss as the criterion of sensitivity experiments. All the results are shown in Fig 3.

– For α, RankingLoss has a obvious inflection point when α changes from 0 to 15. Specially, RankingLoss achieves the best value when α gets 15 and 30. In general, the trend of RankingLoss is gentle when α changes, which shows that our proposed framework is not sensitive to α when its value is not too small.
– For β, RankingLoss gets the best value when β is 0.005 and 0.01. When β increases after 0.01, RankingLoss gets worse obviously.
– For the number of nodes k of the embedding layer, RankingLoss reduces firstly and then increases slightly. It is interesting that RankingLoss gets its best value when k is relatively small, which guarantees that we can speed up the construction of our model because of the low dimension. Moreover, the trend of RankingLoss is gentle when k changes, which is helpful for the tuning process of k.

As a whole, we set $\alpha = 15$, $\beta = 0.005$, $k = 100$ for all datasets according to the parameter sensitivity experiments.

4.6 Effects on Supervision Information

To study the effectiveness of the proposed model in the case there are different numbers of labeled instances are available, we do a series of experiments in variable ratios

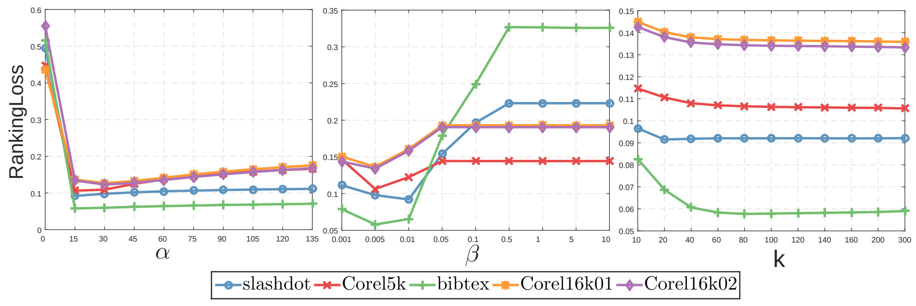

Fig. 3: The Parameter Affects

of labeled instances. Specifically, the ratio of labeled instances increases from 5% to 50% and the step size is 5%. The results are shown in Fig. 4. It is obvious that SERL achieves the best performance in all ratios, which demonstrates the effectiveness of SERL. Moreover, compared to the baselines, the superiority of SERL is higher in small ratios than high ratios, which shows that SERL can make full use of labeled and unlabeled data sufficiently.

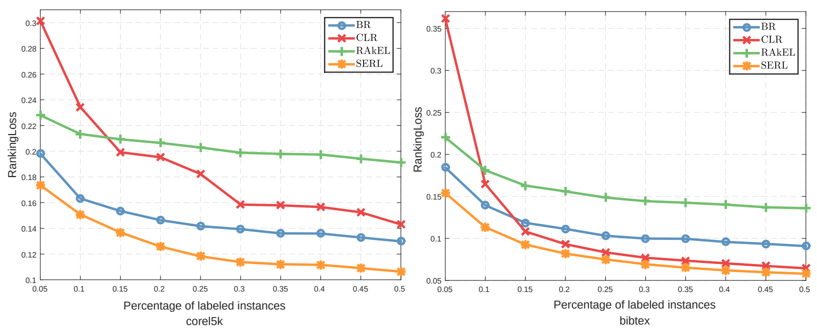

Fig. 4: The Performance in Variable Ratios of Labeled Instances

5 Related Work

Multi-label learning has attracted a lot of interest in recent years. There are two kinds of multi-label algorithms, problem transformation and algorithm adaptation methods.

Problem transformation methods transform multi-label learning problem into other problems which have solid theories and well-established solutions. For example, Binary Relevance [1], AdaBoost.MH [18], Stacked Aggregation [8] and Classifier Chains [16] transform multi-label learning problem into binary classification problems. Calibrated Label Ranking transforms multi-label learning problem into label ranking problems with calibrated scenario by pairwise comparison [6]. Random k-Labelsets [24] transforms multi-label learning problem into multi-class classification problems on an ensemble of k random label subsets.

Algorithm adaptation methods adapt traditional algorithms to multi-label data [31]. For example, ML-kNN [29] adapts traditional k-nearest neighbor algorithm to multi-label data and uses maximum a posteriori(MAP) principle to predict labels for the new instance. ML-DT [4] calculates information gain based on multi-label entropy. Rank-SVM [5] fits the maximum margin to differentiate the relevant and irrelevant labels of one instance. BP-MLL [30] uses feedforward neural network to hold multi-label data where a error function capturing ranking correlation between relevant and irrelevant labels is calculated through backpropagation algorithm. CML [7] utilizes conditional random field to model label co-occurrences in multi-label data. Nguyen et al. [13] proposed a Bayesian nonparametric approach to learn the number of label-feature correlation pat- terns automatically. MLFE [32] utilized the structural information in feature space to enrich the labeling information.

Except these algorithms, representation learning is also one of the most important aspects of multi-label learning [33, 15, 27, 26, 3, 19, 14, 9, 10, 34]. For example, Read et al. [15] utilized restricted boltzmann machine (RBM) to achieve better representation of the original features to train the classifier. BILC [34] mapped the label relationship into a binary embedded space instead of real-valued to achieve better performance. However, current works about representation learning neglect label knowledge, or suffer from the lack of labeled data, or are limited to linear projection. The most related work [11], which proposed a bi-directional representation model for multi-label classification, in which the mid-level representation layer is constructed from both input and output spaces. In essence, their network structure is different from ours. Their framework contained two basic autoencoders, i.e., one for the input features and the other one for output labels, and had to compute the additional parameters of encoding weights from low-dimensional representation of the input features to output labels and prediction model from input features to the low-dimensional representation of the output labels. In this paper, we propose a framework named SERL, which adopts a two-encoding-layer autoencoder to learn feature representation in a supervised manner. The autoencoder can sufficiently utilize labeled and unlabeled data simultaneously under the supervision of softmax regression. The softmax regression incorporates label knowledge to improve the performance of both representation learning and multi-label learning by being jointly optimized with the autoencoder. Extensive experiments on five data sets demonstrate the good performance of our framework.

6 Conclusion

In this paper, we proposed a framework named SERL, which adopts autoencoder to learn feature representation in a supervised manner. In this framework, labeled and unlabeled data can be handled by the autoencoder, meanwhile the softmax regression incorporates label knowledge by being jointly optimized with autoencoder. Moreover, the autoencoder is expanded into two encoding layers to share knowledge with softmax regression by sharing the second encoding weight matrix. Extensive experiments on five real-world datasets demonstrate the superiority of SERL over other state-of-the-art multi-label learning algorithms.

References

1. M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.
2. C. C. Chang and C. J. Lin. Libsvm: A library for support vector machines. *Acm Transactions on Intelligent Systems & Technology*, 2(3):27, 2011.
3. G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *Siam International Conference on Data Mining, SDM 2008, April 24-26, 2008, Atlanta, Georgia, Usa*, pages 410–419, 2008.
4. A. Clare and R. D. King. Knowledge discovery in multi-label phenotype data. *Lecture Notes in Computer Science*, 2168(2168):42–53, 2002.
5. A. E. Elisseeff and J. Weston. A kernel method for multi-labelled classification. *Advances in Neural Information Processing Systems*, 14:681–687, 2002.
6. J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker. Multilabel classification via calibrated label ranking. *Machine learning*, 73(2):133–153, 2008.
7. N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 195–200. ACM, 2005.
8. S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.
9. S. Ji, L. Tang, S. Yu, and J. Ye. A shared-subspace learning framework for multi-label classification. *Acm Transactions on Knowledge Discovery from Data*, 4(2):1–29, 2010.
10. K. Karalas, G. Tsagkatakis, M. Zervakis, and P. Tsakalides. Deep learning for multi-label land cover classification. In *SPIE Remote Sensing*, pages 96430Q–96430Q. International Society for Optics and Photonics, 2015.
11. X. Li and Y. Guo. Bi-directional representation learning for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 209–224. Springer, 2014.
12. D. Madigan, A. Genkin, D. D. Lewis, S. Argamon, D. Fradkin, Y. Li, and D. D. L. Consulting. Author identification on the large scale. *Proc of the Meeting of the Classification Society of North America*, 2005.
13. V. Nguyen, S. Gupta, S. Rana, C. Li, and S. Venkatesh. A bayesian nonparametric approach for multi-label classification. In *Asian Conference on Machine Learning*, pages 254–269, 2016.
14. B. Qian and I. Davidson. Semi-supervised dimension reduction for multi-label classification. In *AAAI*, volume 10, pages 569–574, 2010.
15. J. Read and F. Perezcruz. Deep learning for multi-label classification. *Machine Learning*, 85(3):333–359, 2014.
16. J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333, 2011.
17. J. Read, P. Reutemann, B. Pfahringer, and G. Holmes. Meka: A multi-label/multi-target extension to weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016.
18. R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168, 2000.
19. L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, Usa, August*, pages 668–676, 2008.
20. G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3), 2006.
21. G. Tsoumakas, I. Katakis, and I. Vlahavas. *Mining Multi-label Data*. 2009.
22. G. Tsoumakas, I. Katakis, and I. Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011.

23. G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12(Jul):2411–2414, 2011.
24. G. Tsoumakas and I. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *European Conference on Machine Learning*, pages 406–417. Springer, 2007.
25. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.
26. K. Yu, S. Yu, and V. Tresp. Multi-label informed latent semantic indexing. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 258–265, 2005.
27. M.-L. Zhang and L. Wu. Lift: Multi-label learning with label-specific features. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):107–120, 2015.
28. M.-L. Zhang and K. Zhang. Multi-label learning by exploiting label dependency. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 999–1008. ACM, 2010.
29. M. L. Zhang and Z. H. Zhou. A k-nearest neighbor based algorithm for multi-label classification. In *IEEE International Conference on Granular Computing*, pages 718–721 Vol. 2, 2005.
30. M. L. Zhang and Z. H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge & Data Engineering*, 18(10):1338–1351, 2006.
31. M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.
32. Q.-W. Zhang, Y. Zhong, and M.-L. Zhang. Feature-induced labeling information enrichment for multi-label learning. 2018.
33. Y. Zhang and Z. H. Zhou. Multi-label dimensionality reduction via dependence maximization. In *AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, Usa, July*, pages 1503–1505, 2008.
34. W.-J. Zhou, Y. Yu, and M.-L. Zhang. Binary linear compression for multi-label classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3546–3552. AAAI Press, 2017.