Multi-label Classification with Feature-aware Implicit Encoding and Generalized Cross-entropy Loss

Fatemeh Farahnak-Ghazani
MS student
Computer Engineering
Sharif University of Technology
Tehran, Iran, 1458889694
Email: farahnak@ce.sharif.edu

Mahdieh Soleymani Baghshah
Assistant Professor
Computer Engineering
Sharif University of Technology
Tehran, Iran, 1458889694
Email: soleymani@sharif.edu

Abstract— **In multi-label classification problems, each instance can simultaneously have multiple labels. Since the whole number of available labels in real-world applications tends to be (very) large, the computational cost becomes an important challenge and recently label space dimension reduction (LSDR) methods have received attention. These methods first encode the output space to a low-dimensional latent space. Afterwards, they predict the latent space from the feature space and reconstruct the original output space using a suitable decoding method. The encoding method can be implicit which learns a code matrix in the latent space for the available data or explicit which learns a direct encoding function. It can be feature-aware which considers predictability of the latent space from the feature space or not feature-aware which obtains the latent space from only the label space. In this paper, we propose FIECE method which is a feature-aware implicit encoding method. FIECE uses a generalized cross-entropy loss function for reconstruction error of the label space. Since label vectors in these problems are usually sparse, we use a parameter in the cost function to address the imbalanced classification problem for each label. Extensive experiments on several datasets demonstrate effectiveness of the proposed method compared to some previous methods.**

Keywords- feature-aware encoding; generalized cross-entropy; implicit encoding; label space dimension reduction; multi-label classification

I. INTRODUCTION

Multi-label classification is an important problem in data analysis and has received a lot of attention by researchers especially in the last years. In this problem each instance can simultaneously have multiple labels. For example, an image can have different labels or tags such as “sea”, “boat”, and “beach”. Some applications of multi-label classification are automatic image annotation, video annotation, text classification, gene function prediction, and music genre classification. Until now, many methods have been proposed to solve the multi-label classification problem. Most of these methods learn a mapping from the input space to the output space directly [1], [2], [3], [4], [5], [6]. For example, BR [1] is a simple method which learns independent binary classifiers for labels.

In some recent applications of multi-label classification, the whole number of available labels tends to be (very) large and the label vector of each instance is sparse. Thus, for these applications, the computational cost of methods becomes an important challenge. One of the paradigms presented to tackle multi-label classification problems with a large number of possible labels is the label space encoding. Some recent methods present a max/large margin formulation for this problem and then convert it to a metric learning problem [7], [8]. LM-*k*NN [8] is a scalable state-of-the-art multi-label classification method and learns an encoding matrix which transforms the label space to an embedded space by solving a large margin formulation with constraints on *k* nearest neighbor instances for each instance. Indeed, LM-*k*NN is a scalable extension of the proposed method in [9] in which the constraints for each instance are on all other instances. Another class of label space encoding methods known as label space dimension reduction (LSDR) encode the label space to a low-dimensional latent space [10], [11], [12], [13], [14]. Then, they learn predictive mappings from the feature space to this latent space rather than the original high-dimensional label space at much lower cost. In the prediction phase, they first obtain the code vector of each instance in the latent space based on its feature vector using the learnt mappings. Then, this code vector is decoded to its original label vector. The encoding method is called explicit if there exists a specific encoding function and called implicit if the code matrix (in the latent space) - assuming to be the result of an implicit encoding function - is learnt by optimizing a cost function. It is called feature-aware if it considers the predictability of the latent space from the input space during the learning process (and not feature-aware if it doesn’t consider it). LSDR methods do not attend that the number of zeros for each label is usually much more than ones in multi-label classification problems and each instance has a sparse label vector. Moreover, they usually consider a simple cost function such as Sum of Square Error (SSE) for reconstruction error of the label space while the labels have binary values and SSE is not a proper cost function for this purpose.

In this paper, we propose a feature-aware implicit encoding method with a generalized cross-entropy loss function which is better than sum of square error for classification problems and

handles imbalance between the number of zeros and ones in label vectors.

The remaining of this paper is organized as follows. Sec-tion 2 includes definition of multi-label classification problem and a review of related works. Section 3 presents the proposed method in detail. The setting and results of experiments are included in Section 4. Finally, Section 5 includes the paper conclusion.

II. Related Work

Assume $\{(x_i, y_i)\}_{i=1}^{N}$ is a set of N training instances, where $x_i \in \mathcal{X}$ (and $\mathcal{X} = \mathbb{R}^d$) is the d-dimensional feature vector of the i^{th} instance and $y_i \in \mathcal{Y} = \{0,1\}^Q$ is the corresponding Q-dimensional label vector. Q is the whole number of available labels. The one/zero-valued entries in the label vector of each instance indicate its relevant/irrelevant labels. The data feature matrix is $X_{N\times d} = [x_1, \dots, x_N]^T$ and the label matrix is $Y_{N\times Q} = [y_1, \dots, y_N]^T$. The goal of the multi-label classification problem is to find a mapping $\mathcal{F}: \mathcal{X} \to \mathcal{Y}$ to predict the label vector of a testing instance based on its feature vector.

Most of the proposed methods to solve the multi-label classification problem learn the mapping $\mathcal{F}$ from the input space to the output space directly. Since computational cost is an important challenge in recent applications with large number of available labels, LSDR methods have received attention recently. These methods consider a low dimensional latent space $\mathcal{C} \subset \mathbb{R}^L$ ($L \ll Q$) between the input space and the output space, which can be viewed as an encoding of the output space. Afterwards, they learn the predictive mappings $\mathcal{H}: \mathcal{X} \to \mathcal{C}$ from the input space to this latent space rather than to the original output space. To find the label vector of an unseen instance, they first predict the corresponding latent vector based on its feature vector using $\mathcal{H}$, then use a decoding function $\mathcal{Q}: \mathcal{C} \to \mathcal{Y}$ followed by a thresholding to obtain its label vector.

CS [10] is one of the first methods which uses LSDR and encodes the label vectors using a small number of random projections under the sparsity assumption for label vectors. Then, it uses standard algorithms in the decoding phase to recover the original label vectors. PLST [11] uses Principal Component Analysis (PCA) to transform the label space to a low-dimensional latent space. CPLST [12] combines the cost function of PCA and error of predicting the latent space from the input space using linear regression to propose a feature-aware label space dimension reduction method. FaIE [13] considers a code matrix in a low-dimensional latent space as a result of an implicit encoding function and learns it by optimizing a cost function which maximizes both predictability of the latent space and recoverability of the output space. This method has a feature-aware implicit label space encoding.

Among LSDR methods, those which have feature-aware encoding are more efficient because of considering the predictability of latent space [12], [13] from the input space. Besides those which have implicit encoding can be more effective because no limiting assumption is considered on the encoding function [13]. CPLST and FaIE are state-of-the-art LSDR methods.

III. Proposed Approach

In this section, we propose a feature-aware implicit encoding method which uses a generalized cross-entropy loss function to consider the reconstruction error of labels. This loss function has a parameter that addresses the imbalance of zeros and ones in label vectors.

A. Proposed FIECE

To achieve good performance, LSDR methods need to have both appropriate mapping $\mathcal{H}$ and decoding function $\mathcal{Q}$ [13]. Therefore, the proposed FIECE simultaneously minimizes the prediction error of the latent space and the reconstruction error of the output space to obtain required parameters including the code matrix C and the matrix D used in the decoding phase. For the prediction error of the latent space from the input space, the SSE cost function is used and for the reconstruction error of the label space from the latent space as mentioned earlier a generalized cross-entropy (with a balancing parameter) is used. These two cost functions are combined with each other to obtain the overall one. After solving the optimization problem and learning matrices C and D, a linear ridge regression is utilized to learn a prediction matrix W that is used to find C from X. To predict the label vector of an unseen instance, it first predicts the corresponding latent vector based on the feature vector using the matrix W. Afterwards, it uses the matrix D and a nonlinear function to obtain the real valued label vector and then uses a threshold to obtain its binary valued label vector.

1) Prediction error of the latent space

To predict the code matrix $C_{N\times L}$ from the feature matrix $X_{N\times d}$, the transformation matrix $W_{d\times L}$ is used:

$$\hat{C} = XW \tag{1}$$

The prediction error of reaching the latent space from the feature space denoted as $J_1(C, \hat{C})$ is computed as follows:

$$J_1(C, \hat{C}) = \sum_{i=1}^{N}\sum_{j=1}^{L}\left\| c_i^{(j)} - \hat{c}_i^{(j)} \right\|^2 = \|C - XW\|_F^2 \tag{2}$$

2) Reconstruction error of output space

To decode the label matrix Y from the code matrix C, a transformation matrix D is utilized. To use the cross-entropy loss function as a more desirable loss function for classification problems, a logistic sigmoid function is needed to be applied on the resulting matrix as follows:

$$\hat{Y} = \sigma(\mu CD) \tag{3}$$

where $\sigma(x) = \frac{1}{1+exp(-x)}$ and μ is a parameter that is used to control steepness of the sigmoid function. The reconstruction error of the output based on a generalized cross-entropy loss function is computed as follows:

$$J_2(Y,\hat{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{Q}\Big(\beta\, y_i^{(j)}\, ln\Big(\hat{y}_i^{(j)}\Big) + \big(1-\ y_i^{(j)}\big)\, ln\Big(1-\ \hat{y}_i^{(j)}\Big)\Big) + \gamma\|D\|_F^2 \quad (4)$$

where $y_i^{(j)}$ and $\hat{y}_i^{(j)}$ denote the jth label in the corresponding original and predicted label vector of the i th instance respectively. γ is a regularization parameter that is used as the coefficient of the regularization term on the matrix D and $\|.\|_F$ is the *Frobenius norm* of a matrix. β is balancing parameter which is defined as the proportion of the mean number of zeros in training label vectors and the mean number of ones in this vectors, by:

$$\beta = \frac{\frac{1}{N}\sum_{i=1}^{N}(Q-\|y_i\|_1)}{\frac{1}{N}\sum_{i=1}^{N}\|y_i\|_1} \quad (5)$$

where $\|.\|_1$ denotes the L_1 norm of a vector. Since the number of ones in label vectors in the multi-label classification problems is low, considering this parameter causes equal importance in predicting zeros and ones in label vectors. If we do not consider this parameter, the model tends to predict the zero-valued labels right to cause less error in J_2, and pays less attention to predicting the one-valued labels.

Assuming $C=[c_1,c_2,\dots,c_N]^T$ and $D=[d_1,d_2,\dots,d_Q]$, by substituting (3) in (4), the following cost function is obtained:

$$J_2(Y,\hat{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{Q}\Big(\beta\, y_i^{(j)}\, ln\Big(\sigma\big(\mu c_i^T d_j\big)\Big) + \big(1-\ y_i^{(j)}\big)\, ln\Big(1-\ \sigma\big(\mu c_i^T d_j\big)\Big)\Big) + \gamma\|D\|_F^2 \quad (6)$$

Equation (6) can be written using a new notation as follows:

$$J_2(Y,\hat{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{Q}\Big(\beta\, Y_{i,j}\ln\Big(\sigma\big(\mu C_{i,.}D_{.,j}\big)\Big) + \big(1-\ Y_{i,j}\big)\ln\Big(1-\ \sigma\big(\mu C_{i,.}D_{.,j}\big)\Big)\Big) + \gamma\|D\|_F^2 \quad (7)$$

where for an arbitrary matrix A, $A_{i,j}$ denotes the entry in the ith row and the jth column of the matrix A. $A_{i,.}$ shows the ith row and $A_{.,j}$ denotes the jth column of this matrix.

3) The proposed cost function

By integrating the prediction error of the latent space and the reconstruction error of the output space (with the balancing parameter α for the former), the overall cost function is defined as:

$$J(C,D,W) = \alpha J_1\big(C,\hat{C}\big) + J_2\big(Y,\hat{Y}\big) \quad (8)$$

$$J(C,D,W) = \alpha\|C-XW\|_F^2 - \sum_{i=1}^{N}\sum_{j=1}^{Q}\Big(\beta\, Y_{i,j}\ln\Big(\sigma\big(\mu C_{i,.}D_{.,j}\big)\Big) + \big(1-\ Y_{i,j}\big)\ln\Big(1-\ \sigma\big(\mu C_{i,.}D_{.,j}\big)\Big)\Big) + \gamma\|D\|_F^2 \quad (9)$$

4) Minimizing the cost function

First we obtain the optimum value of the matrix W, which is appeared only in the first statement:

$$\frac{\partial}{\partial W}\|C-XW\|_F^2 = 0 \Rightarrow\ W = (X^TX)^{-1}X^TC \quad (10)$$

By substituting (9) in $J_1\big(C,\hat{C}\big)$ we obtain

$$J_1(C,\hat{C}) = \mathbf{Tr}(C^T(I-\Delta)C) \quad (11)$$

$$\Delta = X(X^TX)^{-1}X^T \quad (12)$$

where $\mathbf{Tr}(.)$ denotes the trace of a matrix. The matrix X^TX is assumed to be invertible which in practice we add a small value ϵ to its diagonal elements to ensure the matrix is invertible.

By replacing (10) in (8), the cost function J will be a function of C and D. Since $J(C,D)$ is nonlinear relative to both matrices, the gradient descent (with learning rate η) is used to learn them in which we need these derivations:

$$\frac{\partial J_1(C,\hat{C})}{\partial C} = 2(I-\Delta)C \quad (13)$$

$$\frac{\partial J_2(C,\hat{C})}{\partial C_{i,k}} = -\sum_{j=1}^{Q}\mu D_{k,j}\Big(\beta\, Y_{i,j}\big(1-\sigma\big(C_{i,.}D_{.,j}\big)\big) - \big(1-\ Y_{i,j}\big)\sigma\big(C_{i,.}D_{.,j}\big)\Big) \quad (14)$$

$$\frac{\partial J_1(C,\hat{C})}{\partial D} = 0 \quad (15)$$

$$\frac{\partial J_2(Y,\hat{Y})}{\partial D_{k,j}} = -\sum_{j=1}^{Q}\mu C_{i,k}\Big(\beta\, Y_{i,j}\big(1-\sigma\big(C_{i,.}D_{.,j}\big)\big) - \big(1-\ Y_{i,j}\big)\sigma\big(C_{i,.}D_{.,j}\big)\Big) + 2\,\gamma D \quad (16)$$

5) Learning regression functions to predict the latent space

After learning the code matrix C, we use a linear ridge regression to learn matrix W to predict the matrix C from the matrix X according to training data, which will be used in the prediction phase. The prediction matrix W can be found as in (10).

6) Prediction phase

To predict the label vector of an unseen instance based on its feature vector, first we predict its code vector c_{test} using prediction matrix W. Then, we use a decoding function with the learnt matrix D and replace C by $[c_{test}]^T$ in (3) to obtain a real valued output vector. Afterwards, we use a threshold to obtain the corresponding label vector with binary values.

B. Kernel Version

Similar to the previous works [12], [13], we use the kernel trick to capture more sophisticated relations between the feature space and the latent space. Assume each feature vector x_i is mapped to a new feature vector θ_i using a mapping function and Θ is the new feature matrix with N rows. $\kappa(x_i, x_j) = \theta_i^T \theta_j$ is the kernel function and $K = \Theta\Theta^T$ is the kernel matrix. According to the representor theorem, the mapping function from the new feature space to the latent space has the form $W' = \Theta^T B$, where $B_{N\times L}$ is the coefficient matrix. In this case, the prediction error of the latent space is as follows:

$$\begin{aligned} J_1(C, \hat{C}) &= \left\| C - \Theta W' \right\|_F^2 \\ &= \|C - \Theta\Theta^T B\|_F^2 \\ &= \|C - KB\|_F^2 \end{aligned} \quad (17)$$

In this formula, we can assume that the feature matrix X is replaced by K and the prediction matrix W is replaced by B compared to the linear case. Thus, the solution for this case is obtained by the same replacements. That means Eq. (12) for calculating Δ which is used in the cost function of linear case, is changed to:

$$\Delta = K(K^T K)^{-1} K^T \quad (18)$$

and all the other equations remain unchanged. Similarly, we add small value ϵ to all entries on the diagonal of the matrix K to make it invertible.

In the prediction phase, we use the kernel ridge regression instead of the linear ridge regression to find the latent space from the input space.

IV. Experiments

We evaluate the performance of the proposed FIECE on a variety of multi-label datasets and compare its results with other methods using three evaluation measurements.

A. Experimental Setup

1) Datasets

The experiments are performed on five multi-label datasets with a relatively large number of possible labels. Table 1 presents properties of these datasets. The first two datasets, *mediamill*, and *cal500* are downloaded from Mulan [15]. The second three datasets, *corel5k*, *espgame*, and *iaprtc12* are prepared image datasets downloaded from [16] for which image features using different methods have been extracted. For these three datasets, we use 100 features extracted using DenseHue method. Table 1 represents properties of these datasets.

TABLE I. Properties of datasets.

Dataset	Domain	# of instances (N)	# of features (d)	# of labels (Q)
mediamill	video	43907	120	101
cal500	music	502	68	174
corel5k	image	4999	100	260
espgame	image	20770	100	268

2) Performance Measures

There are several performance measures to evaluate multi-label classification methods and compare them with each other. We use label-based *micro-F1* and example-based *example-F1* which are the most popular and commonly used measures recently [8].

3) Baseline Methods

To validate our method and compare it with other methods, we take BR [1], LM-*k*NN [8], PLST [11], *kernel*-CPLST [12], FaIE [13] and *kernel*-FaIE.

4) Settings

We report the results of LSDR methods including PLST, CPLST, FaIE, and our proposed FIECE by varying dimension reduction rate:

$$\text{Dimension reduction rate} = L/Q \quad (19)$$

in the range $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. Following the settings in [13], for datasets with more than 5000 instances, we select 5000 instances of them randomly to reduce computational cost. Afterwards, we divide each dataset into 5 parts evenly and randomly and perform 5 runs for each method corresponding to these parts. In each run, one of the parts is considered as a test part and the others are considered as train parts. Then, we average over the results which are obtained using the mentioned evaluation measures and report their mean and standard deviation. For methods that are extended to kernel version, we use the Gaussian kernel and set the smoothing parameter as twice the average of the Euclidean distance between each pair of feature vectors for the corresponding part of the dataset. For the kernel ridge regression, we use functions implemented in [17]. If a method has parameter(s), we divide the training instances into 5 parts randomly and use the first four parts for training and the last one for validation. Then, we consider a range of values for each parameter and use grid search to select the ones which cause the best results on validation data. Other settings for each method are as follows:

- FIECE (our method): The parameter μ used in the decoding phase is set to 0.01. This parameter is used to smooth the output of the decoding function, so the prediction error obtained using the generalized cross-entropy loss function becomes computationally feasible. In other words, it prevents the prediction error from being infinite. The regularization parameter γ is selected from {0.01, 0.05, 0.1, 0.5} and the balancing parameter α is chosen from {0, 0.05, 0.1, 0.5}. The learning rate in gradient descent method is set to 0.1. The small-valued parameter ϵ is set to 10^{-4}.
- BR: We use package LIBLINEAR [18] to learn independent binary classifiers and use L2-regularized logistic regression in the training function. To obtain binary classification results we use 0.5 as a threshold.
- FaIE: The balancing parameter α is selected from $\{10^{-1}, 10^0, \ldots, 10^4\}$ and similar to FIECE the corresponding parameter ϵ is set to 10^{-4}. The code is

received from the authors, but is changed in some cases to be comparable with other methods.

- LM-*k*NN: According to the setting in [8], the number of nearest neighbors is set as $k = 10$. The code is received from the authors.

B. *Experimental results*

The average results of all comparison methods on all five datasets with different dimension reduction rates are represented in terms of *micro-F1* and *example-F1* in Fig. 1 and Fig. 2 respectively. Note that for BR and LM-*k*NN, the results are repeated for different dimension reduction rates to make them comparable with LSDR methods. The experimental results of both linear and kernel version of LSDR methods are reported in

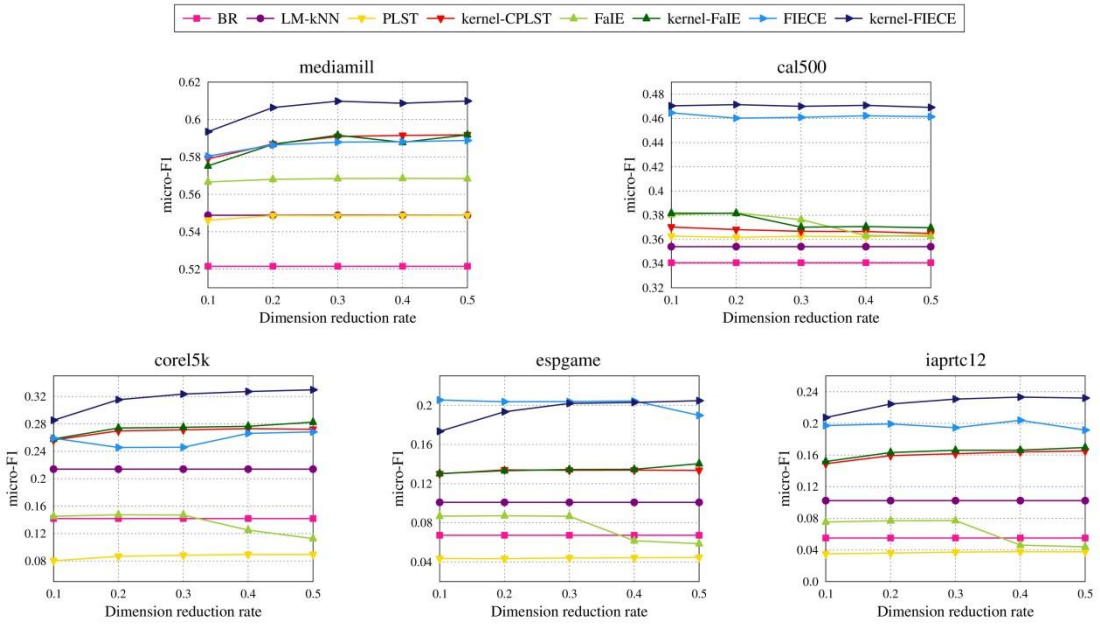

Figure 1. Micro-F1 results of all methods on all datasets with different dimension reduction rates.

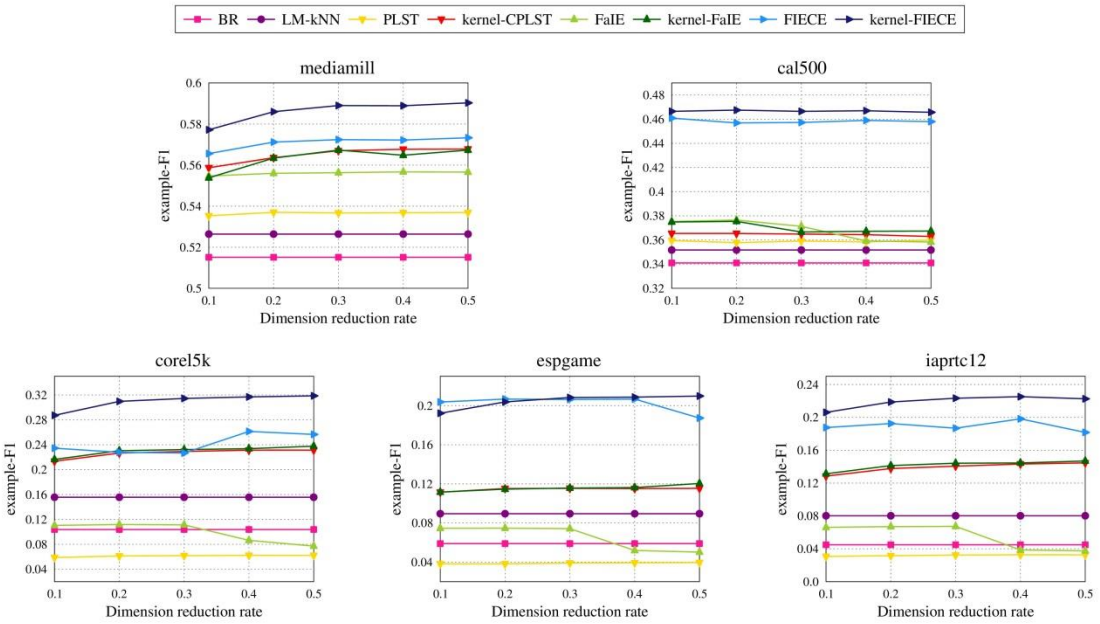

Figure 2. Examole-F1 results of all methods on all datasets with different dimension reduction rates.

the figures. Since we found that linear PLST and linear CPLT have the same results, we reported the results of the earlier method. According to the obtained results, we can see that the linear FaIE outperforms linear PLST on all the datasets. *kernel*-FaIE has similar but slightly better performance than *kernel*-CPLST. Kernel version of both methods yields better performance than their linear versions except on *cal500* which yields similar performance. The comparison results of the two LSDR methods, i.e., FaIE and CPLST, validate the effectiveness of implicit encoding against explicit encoding. The large margin based method LM-*k*NN outperforms FaIE and PLST on all the datasets except on *mediamill* and *cal500*. However, it shows generally an inferior performance compared to the kernel version of the two LSDR methods. *kernel*-CPLST, *kernel*-FaIE and LM-*k*NN outperforms baseline method BR on all the datasets. FaIE shows better performance than BR on all the datasets except on the three image datasets across higher dimension reduction rates. PLST shows better performance than BR on *mediamill* and *cal500* but shows inferior performance on the other datasets. The linear version of the proposed FIECE demonstrates superior performance compared to the other linear LSDR methods, LM-*k*NN, and BR on all the five datasets. *kernel*-FIECE shows the best average results in terms of both performance measures on all the datasets across different dimension reduction rates.

The experimental results demonstrate that using implicit encoding and considering the generalized cross-entropy loss function for the reconstruction error of labels is an effective model, instead of using explicit encoding or SSE cost function for the reconstruction error. Moreover, extending linear LSDR methods to kernel version is an effective paradigm including in our proposed FIECE which yields better results.

V. Conclusion

In this paper, we proposed a label space dimension reduction method FIECE with feature-aware implicit encoding and a generalized cross-entropy loss function. We learned a code matrix and a matrix used in the decoding phase by optimizing a cost function which integrates the prediction error of the latent space and the reconstruction error of the output space. We used the generalized cross-entropy loss to balance the model tendency to truly predict the zero-valued and one-valued labels, while pervious methods do not attend the imbalance between the number of zeros and ones in the label vectors. Moreover, we introduced the kernel version of FIECE to capture more sophisticated relations between the feature space and the latent space.

VI. References

[1] Grigorios Tsoumakas, loannis Katakis, and loannis Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*.: Springer US, 2010, pp. 667-685.

[2] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank, "Classifier chains for multi-label classification," *Machine Learning and Knowledge Discovery in Databases*, 2009, vol.5782, pp. 254-269.

[3] Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker, "Multilabel classification via calibrated label ranking," *Machine Learning*, vol. 73, no. 2, pp. 133-153, Nov. 2008.

[4] André Elisseeff and Jason Weston, "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 681-687.

[5] Yuhong Guo and Dale Schuurmans, "Multi-label classification with output kernels," in *Machine Learning and Knowledge Discovery in Databases*, 2013, vol. 8189, pp. 417-432.

[6] Grigorios Tsoumakas and Ioannis Vlahavas, "Random k-Labelsets for Multilabel Classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 406-417, Sep. 2010.

[7] Yi Zhang and Jeff G. Schneider, "Maximum margin output coding," in *International Conference on Machine Learning (ICML)*, 2012, pp. 1575-1582.

[8] Weiwei Liu and Ivor W. Tsang, "Large margin metric learning for multi-label classification," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[9] Daniel D. Lee and H. Sebastian Seung, "Algorithms for Non-negative Matrix Factorization," *Advances in Neural Information Processing Systems 13*, pp. 556-562, 2001

[10] John Langford, Tong Zhang, Daniel J. Hsu, and Sham M. Kakade, "Multi-label prediction via compressed sensing," in *Advances in Neural Information Processing Systems 22 (NIPS)*, vol. 22, 2009, pp. 772-780.

[11] Farbound Tai and Hsuan-tien Lin, "Multilabel classification with principal label space transformation," *Neural Computation*, vol. 24, no. 9, pp. 2508-2542, Sep. 2012.

[12] Yao-nan Chen and Hsuan-tien Lin, "Feature-aware label space dimension reduction for multi-label classification," in *Advances in Neural Information Processing Systems 25 (NIPS)*, 2012, pp. 1538-1546.

[13] Zijia Lin, Guiguang Ding, Mingqing Hu, and Jianmin Wang, "Multi-label classification via feature-aware implicit label space encoding," in *International Conference on Machine Learning (ICML)*, 2014, pp. 325-333.

[14] Ashish Kapoor, Prateek Jain, and Raajay Viswanathan, "Multilabel classification using bayesian compressed sensing." *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[15] Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, and Ioannis Vlahavas, "Mulan: A java library for multi-label learning," *The Journal of Machine Learning Research*, vol. 12, pp. 2411-2414, 2011.

[16] http://lear.inrialpes.fr/people/guillaumin/data.php

[17] Steven van Vaerenbergh, "Kernel methods for nonlinear identification, equalization and separation of signals. " *Universidad de Cantabria, 2010.* Software available at http://sourceforge.net/projects/kmbox/

[18] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, pp. 1871-1874, 2008. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear