GROUP SENSITIVE CLASSIFIER CHAINS FOR MULTI-LABEL CLASSIFICATION

Jun Huang[1], *Guorong Li*[1], *Shuhui Wang*[2], *Weigang Zhang*[3], *Qingming Huang*[1,2]

[1]Key Lab of Big Data Mining and Knowledge Management,University of Chinese Academy of Sciences
[2]Key Lab of Intelligent Information Processing (CAS), ICT, CAS, Beijing, 100190, China
[3]Harbin Institute of Technology at Weihai, Weihai, 264209, China
{jun.huang, guorong.li, weigang.zhang}@vipl.ict.ac.cn, {wangshuhui, qmhuang}@ict.ac.cn

ABSTRACT

In multi-label classification, labels often have correlations with each other. Exploiting label correlations can improve the performances of classifiers. Current multi-label classification methods mainly consider the global label correlations. However, the label correlations may be different over different data groups. In this paper, we propose a simple and efficient framework for multi-label classification, called Group sensitive Classifier Chains. We assume that similar examples not only share the same label correlations, but also tend to have similar labels. We augment the original feature space with label space and cluster them into groups, then learn the label dependency graph in each group respectively and build the classifier chains on each group specific label dependency graph. The group specific classifier chains which are built on the nearest group of the test example are used for prediction. Comparison results with the state-of-the-art approaches manifest competitive performances of our method.

Index Terms— Multi-Label Classification, Group Sensitive, Classifier Chain, Local Label Correlation

1. INTRODUCTION

In many real applications, one object can be assigned with multiple labels simultaneously. For example, in image annotation, an image can be annotated "sea water" and "sea bird"(see Fig.1(a)). Multi-label classification deals with objects having multiple class labels simultaneously and each object is represented by only one single instance. Multi-label classification has attracted significant attentions from researchers and has been applied to a variety of domains, such as text classification [1, 2, 3], image annotation [4, 5, 6], video annotation [7, 8], bioinformatics [9, 10], social network [11] and music emotions categorization [12, 13, 14].

This work was supported in part by National Basic Research Program of China (973 Program): 2012CB316400 and 2015CB351802, in part by National Natural Science Foundation of China: 61303153, 61025011, 61332016, 61303160, 61390511 and 61202322, in part by 863 program of China: 2014AA015202, in part by China Postdoctoral Science Foundation: 2014T70111 and 2014T70126, and in part by Present Foundation of UCAS.

(a) sea bird and sea water (b) sailing boat and sea water

Fig. 1. Two nature scene image examples

In multi-label classification, labels often have correlations with each other. It has been shown that exploiting label correlations between labels can improve the performances of the classifiers [15, 16, 17, 18]. For example, if one image is annotated with label "sailing boat", it has a high probability to assign label "sea water" to this image (see Fig.1(b)). Current multi-label classification algorithms mainly exploit label correlations globally, by assuming that the label correlations are shared by all the examples. In many applications, however, different examples may share different label correlations.

In Fig.1, "sailing boat" and "sea bird" all have strong correlations with label "sea water". But these strong correlations may be different on different groups of images. The correlation between "sea bird" and "sea water" will be stronger than the correlation between "sailing boat" and "sea water" in those images which are similar to Fig.1(a), and vice versa.

If there are more images like Fig.1(a) than Fig.1(b) in the training data sets, correlation between "sea bird" and "sea water" will be the strong global correlation. In this case, "sea bird" will be annotated to those images like Fig.1(b). Even more, "sea bird" may be assigned with "sailing boat" instead, but both results are incorrect. This impact could be alleviated if we exploit the label correlations which are shared by different groups of similar examples respectively. Ignoring this phenomenon will degrade the performances of multi-label classification models.

To the best of our knowledge, the only work trying to model label correlation locally is ML-LOC [19]. To encode the local influence of label correlations, ML-LOC constructs a LOC (LOcal Correlation) code for each instance and use

this code as additional features for the instance. However, it is difficult to explain the direct connections between LOC codes and the local label dependency structures.

Considering the problems above, we try to exploit the label correlations locally and propose a simple and efficient framework Group sensitive Classifier Chains (GCC) for multi-label classification. We assume that similar examples not only share the same label correlations, but also tend to have similar labels. In the training stage, GCC first augments the original feature space with label space and clusters them into groups. Then, GCC learns the label dependency graph in different groups respectively (*GCC exploits label correlations locally, and the label dependency graph can well illustrate the dependency structures between labels*) and builds the classifier chains based on each learned group specific label dependency graph. In the test stage, GCC finds the nearest group (the test example is similar to the examples in this group and share the the label correlations with them) for one test example, and the group specific classifier chains are used to predict.

2. RELATED WORK

In the past decades, many well-established methods have been proposed to solve multi-label classification problems in various domains. These methods can be divided into two categories [20, 21]: Problem transformation methods and algorithm adaption methods. Problem transformation methods transform multi-label classification problem into one or more single-label classification problems. Algorithm adaption methods modify traditional single-label learning algorithms for multi-label classification directly.

Binary Relevance (BR) [4] is a representative algorithm of problem transformation methods. BR decomposes the multi-label classification problem into L independent binary (one-vs-rest) classification problems, where each binary classification problem corresponds to one label in the label space. BR is simple and straightforward, but it does not consider label correlations. In multi-label learning, however, labels often have correlations with each other.

Classifier Chains (CC) [15] is a novel chaining method that can model label correlations. CC model transforms the multi-label classification problem into a chain of binary classification problems. It involves L binary classifiers, and each binary classifier is trained one by one. Classifier $h_{\pi(i)}$ is trained by using $y_{\pi(1)}, y_{\pi(2)}, ..., y_{\pi(i-1)}$ as augmented features with the original feature space. Here $y_{\pi(1)}, y_{\pi(2)}, ..., y_{\pi(L)}$ is one possible order of L labels, while the label ordering have significant impact on the performance.

ML-LOC [19] exploits label correlations locally. It assumes that the instances can be separated into different groups and each group shares a subset of label correlations. To encode the local influence of label correlations, it constructs a LOC (LOcal Correlation) code for each instance and use this code as additional features for the instance. The classifier is trained with original features and LOC codes. For test examples, LOC codes are unknown and regression models are trained to predict their LOC codes. However, it is difficult to understand the direct connections between LOC codes and the local label dependency structures.

(a) one-to-two (b) two-to-one

Fig. 2. Two simple situations of global label dependency

3. GCC FRAMEWORK

In this section, details of the proposed framework GCC will be presented. First, we give an analysis of local and global label correlations. Then, we present the total framework of GCC and introduce a simple method to model the label dependency for each group of the data set by a DAG structure.

3.1. Preliminaries

In multi-label learning, let $\mathcal{X} \in \mathbb{R}^d$ be the input space with d-dimensional and $\mathcal{Y} = \{y_1, y_2, ..., y_L\}$ be the finite set of L possible labels. $\mathcal{D} = \{(x_i, Y_i)\}_{i=1}^N$ is the training data set with N examples. The i-th object is denoted by a vector with d attribute values $x_i = [x_{i1}, x_{i2}, ..., x_{id}]$, $x_i \in \mathcal{X}$, and $Y_i = [y_{i1}, y_{i2}, ..., y_{iL}]$ is the possible label sets of x_i. Each element $y_{ij} = 1$ if the label y_j is associated with x_i, otherwise $y_{ij} = 0$. k is the number of groups of one data set.

3.2. Local and Global label correlation

Let's take a further step to understand the differences between local and global label correlations. Suppose there are three labels, Fig.2 shows two simple situations of label dependency structures which are composed of two global label dependency correlations.

In Fig.2(a), it shows the global label dependency of these three labels. Label y_2 and y_3 are dependent on y_1. It can be learned that label y_2 and y_3 are conditional independent given y_1, thus the prediction of $\Pr(y_2|y_1, x)$ and $\Pr(y_3|y_1, x)$ will be the same both under global or local situation. However, suppose one test object only belongs to label y_1 and y_2, but label y_3 will also be predicted with a high probability. In this situation, the multi-label classification model may assign irrelevant labels to an unseen object. But this impact can be alleviated if we decompose it into two local label dependency structures.

In Fig.2(b), label y_3 is dependent on y_1 and y_2 globally. In this situation, incorrect prediction of label y_1 or y_2 will affect

the decision of label y_3. This problem may be worse if the label dependency is "one-to-many" or "many-to-one".

One may note that the global label correlations may work better than local label correlations sometimes, e.g., if the example belongs to these three labels simultaneously. When the groups are clustered inappropriately, local correlations may not works. We will show it in experiments (see Sec. 4).

3.3. GCC framework

To exploit label correlations locally, we first cluster the data set into different groups, and learn the label correlations for each group respectively. Then, we build the multi-label classifier chains classification models on the data set by the learned local label correlations of each group. GCC framework is mainly composed of the following four steps.

Cluster the data into groups: The training data is decomposed into k groups ($\{g_i\}_{i=1}^{k}$) by a user defined clustering method m_c. In our experiments, we simply choose *kmeans* as the clustering method, and the similarity is calculated by Euclidean distance. We assume that similar examples not only share the same label correlations, but also tend to have similar labels, so the label space is augmented as features with the original feature space in the clustering stage of GCC.

Learn the label dependency graph: In this paper, we try to model the label dependency of each group by a DAG structure. The dependency probabilities between labels are simply modeled by the co-occurrence of each pair of labels. The dependency probability of label y_j on y_l is calculated by Eq.(1), it indicates when y_l happened, the probability of y_j to be happened.

$$\Pr(y_j|y_l) = \frac{\vec{y_j}^T \vec{y_l}}{\|\vec{y_l}\|_1} \tag{1}$$

where $\vec{y_l} = [y_{1l}^g, y_{2l}^g, ..., y_{n_g l}^g]^T$, $y_{il}^g \in \{0, 1\}$ is the l-th value in Y_i of x_i in the g-th group, n_g is the number of examples in the g-th group.

There will be two possible links between each pair of labels, and we just retain the link with larger dependency probability and remove another one. Then, we can build k directed label dependency graph on each group of the data set. These graphs may have cycles, however, we need to remove the link with minimum dependency probability in each cycle. After these two pre-processing stages, k DAG label dependency graphs ($\{G_i\}_{i=1}^{k}$) can be obtained.

Build the multi-label classifier chains: In this stage, we build k multi-label classifier chains classification models ($H_i(x)_{i=1}^{k}$) based on each learned label dependency graph G_i and the training data $\mathcal{D}$.

$$H_i(x) = \{h_{i1}(x), h_{i2}(x), ..., h_{iL}(x)\} \tag{2}$$

where each binary classifier h_{il} is defined as:

$$h_{il}(x) = \Pr(y_l|\text{Pa}(y_l, G_i), x) \tag{3}$$

Algorithm 1 GCC Framework

Input:

$\mathcal{D}$: the multi-label training data set, $\mathcal{D} = \{(x_i, Y_i)\}_{i=1}^{N}$;
k : the number of groups;
m_c : the method of clustering;
m_g : the method to construct label dependency graph;
m_h : the method of base classifier;
x_t : a test example;

Output:

$\hat{Y}_t$: the set of predicted labels for x_t;

1: $\{g_i\}_{i=1}^{k} = \text{Cluster}(\mathcal{D}, k, m_c)$;
2: **for** $i = 1$ to k **do**
3: $G_i = \text{LearnDependencyGraph}(\mathcal{D}, g_i, m_g)$;
4: $H_i = \text{BuildClassifier}(\mathcal{D}, G_i, m_h)$;
5: **end for**
6: find the nearest group g_n of x_t;
7: **return** $\hat{Y}_t = H_n(x_t)$;

where $\text{Pa}(y_l, G_i)$ represents the set of parents labels of label y_l in graph G_i. Each binary classifier h_{il} is trained by using the parent labels of y_l in graph G_i as augmented features with the original feature space.

One may note that the dependency structure of some labels may be the same in these k DAG graphs, which means the binary classifiers for these labels need to be trained only once. However, the classifiers for group specific label dependency structures need to be trained more times (at most k times)

Predict: Since we assume that similar examples share the same label correlations, we find the nearest group g_n of the test example x_t by calculating Euclidean distance. We assume that x_t share the same label correlations with the examples in group g_n. Classifier chains H_n is built by the label dependency graph G_n which is learned on group g_n. H_n is used for predict of x_t, and we expect that H_n can predict x_t better than other classifier chains which are learned on other group specific label correlations.

One should note that the testing order of these L binary classifiers in H_n should according to one topological sort order of graph G_n. Because when predicting y_l, it's parent labels are augmented with x_t as features, so the parent labels of y_l should be predicted first. While in the training stage, the ground truth labels are given, the training procedures of these L binary classifier chains can be parallelized. All the procedures of GCC are summarised in Algorithm 1.

4. EXPERIMENTS

4.1. Evaluation Metrics

To evaluate the performance of different algorithms for multi-label classification, we use four common evaluation metrics in [11, 15, 20, 21, 22] to verify the performance. Given a test data set $\mathcal{T} = \{(x_i, Y_i)\}_{i=1}^{m}$, where $Y_i \in \{0, 1\}^L$ is the ground

Table 1. Compared methods

Method	Type of Correlation	Publication
BSVM	none	[4]
CC	global correlation	[15]
BCC	global correlation	[16]
ML-LOC	local correlation	[19]
GCC	local correlation	this paper

truth labels of the i-th example, and $\hat{Y}_i$ is the predicted labels.

- **Hamming loss** [11] evaluates how many times an example-label pair is misclassified, i.e., a label not belonging to the example is predicted or a label belonging to the example is not predicted.

$$\text{Hamming loss} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{L}\sum_{l=1}^{L}[\![Y_{il} \neq \hat{Y}_{il}]\!] \tag{4}$$

  The smaller the value of Hamming loss, the better performance of the classifier.

- **Accuracy** [15] evaluates Jaccard similarity between the ground truth labels and the predicted labels.

$$\text{Accuracy} = \frac{1}{m}\sum_{i=1}^{m}\frac{|Y_i \wedge \hat{Y}_i|}{|Y_i \vee \hat{Y}_i|} \tag{5}$$

- **Exact-Match** [15] evaluates how many times the ground truth labels and the predicted labels are exactly matched.

$$\text{Exact-Match} = \frac{1}{m}\sum_{i=1}^{m}[\![Y_i = \hat{Y}_i]\!] \tag{6}$$

- **Macro F_1**[11] is the integrated version of precision and recall for each label.

$$\text{Macro } F_1 = \frac{1}{L}\sum_{i=1}^{L}\frac{2p_i r_i}{p_i + r_i} \tag{7}$$

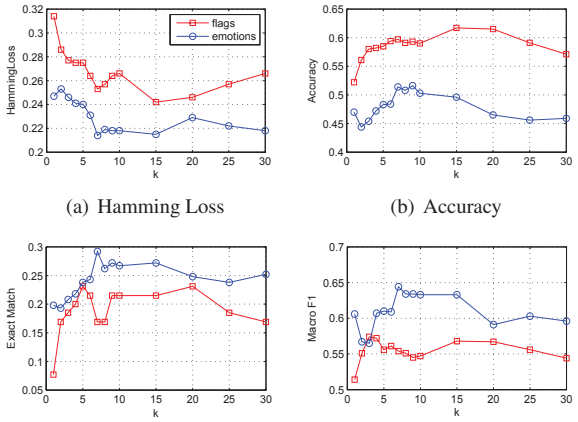
  where p_i and r_i are the precision and recall for the i-th label. For the later four metrics, the larger the value, the better the performance of the classifier.

4.2. Compared methods

We compare the GCC with four state-of-the art well established multi-label learning algorithms: BSVM [4], CC [15], BCC [16] and ML-LOC [19]. For fair comparison, Libsvm (with linear kernel and default parameters) [23] is employed as the base classifier for all the compared algorithms. Parameters for each compared algorithm are suggested by the corresponding publication. For GCC, the number of groups for each data set is set to be 5. All compared algorithms are summarized in Table 1, "Type of Correlation" indicates whether the corresponding method considered the label correlation and which type of correlation they try to exploit.

Table 2. Description of data sets

Data sets	Instance	Features	Labels	Domain
flags	194	19	7	image
image	2000	294	5	image
scene	2407	294	6	image
pascal07	9963	512	20	image
mediamill	43907	120	101	video
emotions	593	72	6	music



(a) Hamming Loss (b) Accuracy

(c) Exact Match (d) Macro F_1

Fig. 3. Results of GCC under different number of groups

4.3. Experiment results and analysis

We experiment on six data sets, the detailed characteristics of these data sets are summarized in Table 2. Most of these data sets can be downloaded from mulan[1] and lamda[2]. Pascal07 [24] with 512-dimensional gist features is download from LEAR [3].

Five-fold cross-validation is performed on each experimental data set. Tables 3 to 6 report the detailed results in terms of different evaluation metrics, and the best result in each row is marked in bold.

We can see that these algorithms (CC, BCC, ML-LOC and GCC) which consider label correlations obtain better performance than BSVM which trains each binary classifier independently. It indicates that exploiting label correlation can improve the performance of classifier. We can also see that these algorithms (ML-LOC and GCC) which exploit label correlation locally can obtain better performance than those algorithms (CC and BCC) which exploit label correlation globally in terms of each evaluation criterion in most cases. These results demonstrate that exploiting the group specific label correlations and building the multi-label classifiers on it can work better. We note that GCC performs better than ML-

[1] http://mulan.sourceforge.net/datasets.html
[2] http://lamda.nju.edu.cn/Data.ashx#data
[3] http://lear.inrialpes.fr/index.php

Table 3. Experiment result (mean±std) of each algorithm in terms of Hamming Loss

Dataset	BSVM	CC	BCC	ML-LOC	GCC
emotions	0.202±0.019	0.214±0.022	0.203±0.022	0.200±0.024	**0.181±0.007**
flags	0.279±0.044	0.287±0.049	0.277±0.030	0.262±0.039	**0.260±0.037**
image	0.175±0.006	0.188±0.006	0.169±0.011	**0.152±0.010**	0.169±0.003
scene	0.109±0.006	0.102±0.007	0.109±0.004	**0.073±0.004**	0.078±0.004
pascal07	0.066±0.001	0.082±0.004	0.066±0.001	**0.063±0.001**	0.069±0.002
mediamill	**0.031±0.000**	0.031±0.001	0.031±0.004	0.032±0.001	0.031±0.001

Table 4. Experiment result (mean±std) of each algorithm in terms of Accuracy

Dataset	BSVM	CC	BCC	ML-LOC	GCC
emotions	0.552±0.033	0.547±0.041	0.547±0.038	0.495±0.075	**0.588±0.008**
flags	0.548±0.056	0.564±0.073	0.576±0.039	0.581±0.062	**0.607±0.052**
image	0.436±0.025	0.566±0.015	0.509±0.018	0.526±0.032	**0.607±0.006**
scene	0.618±0.017	0.705±0.023	0.633±0.024	0.694±0.011	**0.730±0.005**
pascal07	0.189±0.008	**0.293±0.009**	0.193±0.006	0.197±0.009	0.278±0.008
mediamill	0.412±0.003	0.417±0.011	0.402±0.010	0.419±0.009	**0.425±0.001**

Table 5. Experiment result (mean±std) of each algorithm in terms of Exact Match

Dataset	BSVM	CC	BCC	ML-LOC	GCC
emotions	0.282±0.043	0.309±0.053	0.301±0.063	0.279±0.069	**0.363±0.023**
flags	0.145±0.055	0.176±0.075	0.174±0.062	0.179±0.100	**0.263±0.074**
image	0.359±0.034	0.478±0.019	0.417±0.020	0.443±0.029	**0.511±0.016**
scene	0.523±0.022	0.654±0.027	0.560±0.025	0.665±0.014	**0.678±0.008**
pascal07	0.121±0.009	**0.210±0.012**	0.124±0.008	0.118±0.004	0.190±0.007
mediamill	0.087±0.001	0.106±0.013	0.105±0.014	0.105±0.002	**0.116±0.010**

Table 6. Experiment result (mean±std) of each algorithm in terms of Macro-F_1

Dataset	BSVM	CC	BCC	ML-LOC	GCC
emotions	**0.661±0.024**	0.647±0.034	0.649±0.031	0.612±0.061	0.656±0.029
flags	**0.642±0.052**	0.634±0.069	0.561±0.045	0.588±0.053	0.639±0.032
image	0.547±0.013	0.583±0.015	0.596±0.030	0.623±0.023	**0.636±0.006**
scene	0.703±0.017	0.725±0.019	0.697±0.017	0.774±0.012	**0.781±0.008**
pascal07	0.111±0.008	0.170±0.013	0.113±0.006	0.095±0.004	**0.180±0.004**
mediamill	0.107±0.003	0.104±0.006	0.084±0.002	0.064±0.002	**0.108±0.003**

LOC, but it is worse than CC in terms of Accuracy and Exact Match on pascal07. If the number of groups is set inappropriately, global label dependency structure will be decomposed into several unwilling local structures. We can tune the parameter k to achieve a better performance.

4.4. The number of groups

The number of groups k is an important parameter of our GCC framework. To evaluate the influence of parameter k, we perform experiment on flags and emotions with different value of $k \in \{1, 2, ..., 9, 10, 15, 20, 25, 30\}$. For simplicity, we only train and test GCC on the training and testing parts of these two data sets. Experiment results are shown in Fig. 3. We can see that as the number of groups k becomes larger, the performance of GCC first increases then decreases. This is because a large k will make the label correlations too locally. We can obtain a good performance at a relative large k, but the computation will be more expensive with the increasing of k. It is a tradeoff between computation and performance. Specially, GCC learns a global label dependency graph when k=1, but the results of it are almost worse than those results when $k > 1$.

5. CONCLUSIONS

In this paper, we propose a simple framework Group sensitive Classifier Chains for multi-label classification by exploiting label correlation locally. Most of current approaches can be applied to our GCC. We analyzed the differences between global and local label correlations. The empirical experimental results have show that GCC can work better than these methods which do not consider label correlation or exploit label correlation globally. We also show the influence of the number of groups. In the future, we'd like to apply more advanced clustering and label dependency DAG structure learning methods to GCC.

6. REFERENCES

[1] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda, "Maximal margin labeling for multi-topic text categorization," in *NIPS*, 2005, pp. 649–656.

[2] N. Ueda and K. Saito, "Parametric mixture models for multi-labeled text," in *NIPS*, 2003, pp. 721–728.

[3] K. Yu, S. Yu, and V. Tresp, "Multi-label informed latent semantic indexing," in *SIGIR*, 2005, pp. 258–265.

[4] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[5] Y. Luo, D. Tao, C. Xu, D. Li, and C. Xu, "Vector-valued multi-view semi-supervised learning for multi-label image classification," in *AAAI*, 2013, pp. 647–653.

[6] F. Sun, J. Tang, H. Li, G. Qi, and T. S. Huang, "Multi-label image categorization with sparse factor representation," *IEEE Trans. Image Processing*, vol. 23, no. 3, pp. 1028–1037, 2014.

[7] F. Kang, R. Jin, and R. Sukthankar, "Correlated label propagation with application to multi-label learning," in *CVPR*, 2006, pp. 1719–1726.

[8] G. Qi, X. Hua, Y. Rui, J. Tang, T. Mei, and H. Zhang, "Correlative multi-label video annotation," in *ACM Multimedia*, 2007, pp. 17–26.

[9] A. Elisseeff and W. Jason, "A kernel method for multi-labelled classification," in *NIPS*, 2001, pp. 681–687.

[10] X. Wang and G. Li, "Multilabel learning via random label selection for protein subcellular multilocations prediction," *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 10, no. 2, pp. 436–446, 2013.

[11] X. Wang and G. Sukthankar, "Multi-label relational neighbor classification using social context features," in *ACM SIGKDD*, 2013, pp. 464–472.

[12] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. P. Vlahavas, "Multi-label classification of music into emotions," in *ISMIR*, 2008, pp. 325–330.

[13] A. Wieczorkowska, P. Synak, and Z. W. Raś, "Multi-label classification of emotions in music," in *IIS:IIPWM*, 2006, pp. 307–315.

[14] B. Wu, E. Zhong, A. Horner, and Q. Yang, "Music emotion recognition by multi-label multi-layer multi-instance multi-view learning," in *ACM Multimedia*, 2014, pp. 117–126.

[15] J. Read, P. Bernhard, H. Geoff, and F. Eibe, "Classifier chains for multi-label classification," in *ECML-PKDD*, 2009, pp. 254–269.

[16] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, and P. Larrañaga, "Bayesian chain classifiers for multi-dimensional classification," in *IJCAI*, 2011, pp. 2192–2197.

[17] K. Dembczyński, W. Cheng, and E. Hüllermeier, "Bayes optimal multilabel classification via probabilistic classifier chains," in *ICML*, 2010, pp. 1609–1614.

[18] J. Read, L. Martino, and D. Luengo, "Efficient monte carlo methods for multi-dimensional learning with classifier chains," *Pattern Recognition*, vol. 47, no. 3, pp. 1535 – 1546, 2014.

[19] S. J. Huang and Z. H. Zhou, "Multi-label learning by exploiting label correlations locally," in *AAAI*, 2012.

[20] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," *Data Mining and Knowledge Discovery Handbook*, pp. 667–685, 2010.

[21] M. Zhang and Z. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, 2014.

[22] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hllermeier, "Optimizing the f-measure in multi-label classification: Plug-in rule approach versus structured loss minimization," in *ICML*, 2013, pp. 1130–1138.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, 2011.

[24] M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification," in *CVPR*, 2010, pp. 902 – 909.