Multi-View Partial Multi-Label Learning with Graph-Based Disambiguation

Ze-Sen Chen,[1,2] **Xuan Wu,**[3] **Qing-Guo Chen,**[3] **Yao Hu,**[3] **Min-Ling Zhang**[1,2,4*]
[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
[3]YouKu Cognitive and Intelligent Lab, Alibaba Group, Hangzhou, China
[4]Collaborative Innovation Center of Wireless Communications Technology, China
{chenzs, zhangml}@seu.edu.cn, {wx193834, qingguo.cqg, yaoohu}@alibaba-inc.com

Abstract

In multi-view multi-label learning (MVML), each training example is represented by different feature vectors and associated with multiple labels simultaneously. Nonetheless, the labeling quality of training examples is tend to be affected by annotation noises. In this paper, the problem of *multi-view partial multi-label learning* (MVPML) is studied, where the set of associated labels are assumed to be candidate ones and only partially valid. To solve the MVPML problem, a two-stage graph-based disambiguation approach is proposed. Firstly, the ground-truth labels of each training example are estimated by disambiguating the candidate labels with fused similarity graph. After that, the predictive model for each label is learned from embedding features generated from disambiguation-guided clustering analysis. Extensive experimental studies clearly validate the effectiveness of the proposed approach in solving the MVPML problem.

Introduction

The task of multi-view multi-label learning (MVML) widely exists in real-world applications, where each object consists of diverse representations and multiple class labels simultaneously (Luo et al. 2013; Liu et al. 2015; Zhu, Li, and Zhang 2016; Xing et al. 2018; Zhang et al. 2018; Zhu et al. 2018; Wu et al. 2019). For instance, a news webpage can be represented from different views including *text*, *image* and *video*, while at the same time annotated with multiple class labels such as *sports*, *economic*, and *entertainment*.

In conventional MVML studies, it is commonly assumed that all relevant labels have been precisely annotated for each training instance. Nonetheless, in many real-world scenarios, precise annotations are usually difficult and costly to be obtained. As shown in Figure 1, a news webpage with *text*, *image* and *video* views might be annotated with six candidate labels from crowdsourced labelers, among which only *Soccer*, *Europa league* and *England* are ground-truth ones. Under these circumstances, the multiple class labels associated with each training example are only *candidate* ones which are partially valid. In this paper, we formalize the corresponding learning task as the problem of *Multi-View Partial Multi-label Learning* (MVPML).

Figure 1: An examplar multi-view partial multi-label scenario. The news webpage can be represented from different views such as *text*, *image* and *video*. Furthermore, among the six candidate labels annotated by crowdsourced labelers, only three of them are valid ones including *Soccer*, *Europa league* and *England*.

Let $\mathcal{X} = \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \cdots \times \mathbb{R}^{d_V}$ be the feature space consisting of V views, where d_v $(1 \leq v \leq V)$ denotes the dimensionality of the v-th view. Furthermore, let $\mathcal{Y} = \{y_c\}_{c=1}^{q}$ be the label space consisting of q possible class labels. Given the MVPML training set $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i) \mid 1 \leq i \leq m\}$, where $\boldsymbol{x}_i = [\boldsymbol{x}_i^1, \boldsymbol{x}_i^2, \cdots, \boldsymbol{x}_i^V] \in \mathcal{X}$ is the $\sum_{v=1}^{V} d_v$-dimensional feature vector and $Y_i \subseteq \mathcal{Y}$ is the candidate label set associated with $\boldsymbol{x}_i$. The basic assumption of MVPML lies in that the ground-truth labels $\widetilde{Y}_i \subseteq \mathcal{Y}$ for $\boldsymbol{x}_i$ reside in its candidate label set, i.e. $\widetilde{Y}_i \subseteq Y_i$, which are not directly accessible to the learning algorithm. The task of MVPML is to learn a predictive model $h : \mathcal{X} \to 2^{\mathcal{Y}}$ from $\mathcal{D}$ which can assign a set of proper labels for the unseen instance.

Intuitively, the problem of MVPML can be solved by resorting to its degenerated version. Specifically, by ignoring labeling noises in candidate label set and thus treating all candidate labels as ground-truth ones, the MVPML problem will degenerate to the MVML counterpart (Liu et al. 2015; Zhu, Li, and Zhang 2016; Xing et al. 2018; Zhang et al.

*Corresponding author

2018; Wu et al. 2019). Although it is possible to directly invoke existing techniques for MVML to solve the MVPML problem, the resulting performance would be suboptimal as the instrinsic properties of MVPML, i.e. *multi-view* and *noisy labeling*, haven't been fully considered in this way.

In this paper, a first attempt named GRADIS, i.e. *GRAph-based DISambiguation for multi-view partial multi-label learning*, is proposed to solving the MVPML problem. Specifically, GRADIS tackles the noisy labeling of MVPML training examples in two-stage by exploiting the multi-view represetation. In the first stage, GRADIS disambiguates the candidate label set of each training example by conducting label propagation over fused similarity graph. In the second stage, clustering analysis guided by the disambiguation results is performed to help generate embedding features for predictive model induction. Comparative experiments over benchmark data sets clearly validate the effectiveness of GRADIS in learning from MVPML examples.

The rest of this paper is organized as follows. In Section 2, technical details of the proposed GRADIS approach are presented. In Section 3, experimental results of comparative studies are reported. In Section 4, related works on MVPML are briefly discussed. Finally, Section 5 concludes this paper.

The Proposed Approach

To learn from MVPML examples, GRADIS works in two stages including *Candidate Labels Disambiguation* and *Disambiguation-Guided Model Induction*, whose technical details are scrutinized as follows.

Candidate Labels Disambiguation

In the first stage, GRADIS aims to disambiguate the candidate label set of each MVPML training example by conducting graph-based label propagation. Following the same notations in previous Section, given two multi-view instances $\boldsymbol{x}_i = [\boldsymbol{x}_i^1, \boldsymbol{x}_i^2, ..., \boldsymbol{x}_i^V]$ and $\boldsymbol{x}_j = [\boldsymbol{x}_j^1, \boldsymbol{x}_j^2, ..., \boldsymbol{x}_j^V]$, let $\mathcal{N}(\boldsymbol{x}_i^v)$ $(1 \le v \le V)$ denote the set of k-nearest neighbors for $\boldsymbol{x}_i^v$ identified in $\mathcal{D}$ w.r.t. the v-th view. Then, the similarity between two instances w.r.t. the v-th view is calculated as follows:[1]

$$S_{i,j}^v = \begin{cases} e^{-\frac{||\boldsymbol{x}_i^v - \boldsymbol{x}_j^v||^2}{2\sigma^2}}, & \text{if } \boldsymbol{x}_j^v \in \mathcal{N}(\boldsymbol{x}_i^v) \text{ and } i \neq j \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Then, the similarity between instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ can be obtained by fusing similarities from different views:

$$\forall\, 1 \le i, j \le m: \quad W_{i,j} = \sum\nolimits_{v=1}^{V} S_{i,j}^v \quad (2)$$

Accordingly, a weighted directed graph $\mathcal{G} = (V, E, \mathbf{W})$ for the MVPML training set $\mathcal{D}$ is formed by GRADIS. Here, the vertex set $V = \{\boldsymbol{x}_i \mid 1 \le i \le m\}$ consists of all the training instances. Furthermore, the edge set $E = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid 1 \le i, j \le m,\ i \neq j\}$ consists of directed edges between any pair of training instances, and the weight matrix $\mathbf{W} = [W_{i,j}]_{m \times m}$ stores the corresponding similarity values.

[1] In this paper, k-nearest neighbors w.r.t. v-th view are identified using Euclidean distance. Furthermore, the parameter σ in Eq.(1) is fixed to be 1.

Following the label propagation procedure, let $\mathbf{H} = \mathbf{W}\mathbf{D}^{-1}$ be the propagation matrix by normalizing weight matrix $\mathbf{W}$ in column, where $\mathbf{D} = \mathrm{diag}[d_1, d_2, \ldots, d_m]$ is the diagonal matrix with $d_j = \sum_{i=1}^{m} W_{i,j}$. In addition, let $\mathbf{F} = [f_{i,c}]_{m \times q}$ denote the matrix whose entry $f_{i,c} \ge 0$ represents the labeling confidence of y_c being a valid class label for $\boldsymbol{x}_i$. Specifically, the initial labeling confidence matrix $\mathbf{F}^{(0)}$ is set as:

$$\forall\, 1 \le i \le m: \quad f_{i,c}^{(0)} = \begin{cases} \frac{1}{|Y_i|}, & \text{if } y_c \in Y_i \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

For the t-th iteration, $\mathbf{F}$ is updated by propagating current labeling confidence over $\mathbf{H}$:

$$\mathbf{F}^{(t)} = \alpha \cdot \mathbf{H}^\top \mathbf{F}^{(t-1)} + (1 - \alpha) \cdot \mathbf{F}^{(0)} \quad (4)$$

Here, the balancing parameter $\alpha \in [0, 1]$ controls the labeling information inherited from iterative label propagation and the initial labeling confidence $F^{(0)}$. Let $\mathbf{F}^*$ be the final labeling confidence matrix returned by the iterative label propagation procedure[2], which is further re-scaled into $\widehat{\mathbf{F}}$ by normalization w.r.t. the candidate label set:

$$\forall\, 1 \le i \le m: \quad \widehat{f}_{i,c} = \begin{cases} \frac{f_{i,c}^*}{\sum_{y_l \in Y_i} f_{i,l}^*}, & \text{if } y_c \in Y_i \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

Therefore, for each MVPML training example $(\boldsymbol{x}_i, Y_i)$, its candidate label set Y_i is disambiguated into $\widehat{Y}_i$ by thresholding $\widehat{\mathbf{F}}$ w.r.t. parameter $\gamma \in (0, 1)$:

$$\forall\, 1 \le i \le m: \quad \widehat{Y}_i = \{y_c \mid \widehat{f}_{i,c} > \gamma,\ 1 \le c \le q\} \quad (6)$$

Disambiguation-Guided Model Induction

In the second stage, GRADIS aims to induce the predictive model by exploiting the disambiguated training set $\widehat{\mathcal{D}} = \{(\boldsymbol{x}_i, \widehat{Y}_i) \mid 1 \le i \le m\}$. Specifically, the relevancy of each class label is determined by utilizing embedding features generated via clustering analysis.

For the c-th class label y_c, let $\mathcal{I}_c^+$ ($\mathcal{I}_c^-$) denote the index set of training examples which have positive (negative) label assignment w.r.t. y_c:

$$\begin{aligned} \mathcal{I}_c^+ &= \{i \mid y_c \in \widehat{Y}_i,\ 1 \le i \le m\} \\ \mathcal{I}_c^- &= \{i \mid y_c \notin \widehat{Y}_i,\ 1 \le i \le m\} \end{aligned} \quad (7)$$

Then, GRADIS generates a feature embedding $\Phi_c : \mathcal{X} \to \mathcal{Z}_c$ with $\mathcal{Z}_c = \mathcal{Z}_c^1 \times \mathcal{Z}_c^2 \cdots \times \mathcal{Z}_c^V$ for model induction w.r.t. y_c, where $\mathcal{Z}_c^v$ $(1 \le v \le V)$ is instantiated by conducting clustering analysis over the positive (negative) instances indexed by $\mathcal{I}_c^+$ ($\mathcal{I}_c^-$). Clustering analysis serves as a natural way to explore the underlying structure of training examples, which has been successfully utilized to help generate features with strong discriminative ability for multi-label learning (Zhang and Wu 2015; Huang et al. 2016; Weng et al. 2018).

[2] In this paper, the iterative procedure terminates when $\mathbf{F}^{(t)}$ does not change or the maximum number of iterations (i.e. 30) is reached.

Table 1: The pseudo-code of GRADIS.

Inputs:

$\mathcal{D}$: MVPML training set $\{(\boldsymbol{x}_i, Y_i) \mid 1 \le i \le m\}$ $(\boldsymbol{x}_i \in \mathcal{X}, Y_i \subseteq \mathcal{Y}, \mathcal{X} = \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \cdots \times \mathbb{R}^{d_V}, \mathcal{Y} = \{y_1, y_2, \ldots, y_q\})$

k: # nearest neighbors considered in Eq.(1)

α, γ, η: the balancing parameter (Eq.(4)), thresholding parameter (Eq.(6)), and ratio parameter (Eq.(8)) in (0,1)

$\mathcal{B}$: binary training algorithm

$\boldsymbol{u}$: unseen instance

Outputs:

Y: predicted label set for $\boldsymbol{u}$

Process:

1: **for** $v = 1$ to V **do**
2: Set $S^v_{i,j}$ $(1 \le i, j \le m)$ according to Eq.(1);
3: **end for**
4: Form the weighted directed graph $\mathcal{G} = (V, E, \mathbf{W})$ with $\mathbf{W} = [W_{i,j}]_{m \times m}$ and $W_{i,j} = \sum_{v=1}^{V} S^v_{i,j}$;
5: Initialize labeling confidence matrix $\mathbf{F}^{(0)}$ according to Eq.(3);
6: Perform iterative label propagation according to Eq.(4) and return the final labeling confidence matrix $\mathbf{F}^*$;
7: Obtain the disambiguated training set $\widehat{\mathcal{D}} = \{(\boldsymbol{x}_i, \widehat{Y}_i) \mid 1 \le i \le m\}$ with $\widehat{Y}_i$ set according to Eq.(6);
8: **for** $c = 1$ to q **do**
9: Identify the positive and negative index sets $\mathcal{I}^+_c, \mathcal{I}^-_c$ according to Eq.(7);
10: Conduct spectral clustering along with $\mathbf{W}_{\mathcal{I}^+_c}$ and $\mathbf{W}_{\mathcal{I}^-_c}$ to obtain two sets of clustering centers $\{\boldsymbol{p}^1_c, \boldsymbol{p}^2_c, \ldots, \boldsymbol{p}^{m_c}_c\}$ and $\{\boldsymbol{n}^1_c, \boldsymbol{n}^2_c, \ldots, \boldsymbol{n}^{m_c}_c\}$ respectively;
11: Form the binary training set $\widehat{D}_c$ according to Eq.(11), with the feature embedding Φ_c generated w.r.t. Eqs.(9)-(10);
12: Induce binary classifier $f_c \leftarrow\!\shortmid \mathcal{B}(\widehat{D}_c)$;
13: **end for**
14: Return Y according to Eq.(12);

Based on the fused similarity graph $\mathcal{G}$, the widely-used spectral clustering (von Luxburg 2007) is employed to fulfill the clustering task. Let $\mathbf{W}_{\mathcal{I}^+_c}$ denote the weight matrix derived from $\mathbf{W}$ by retaining the rows and columns specified by $\mathcal{I}^+_c$, a set of m_c clustering centers $\{\boldsymbol{p}^1_c, \boldsymbol{p}^2_c, \ldots, \boldsymbol{p}^{m_c}_c\}$ can be returned by invoking the spectral clustering procedure along with $\mathbf{W}_{\mathcal{I}^+_c}$. Here, each clustering center $\boldsymbol{p}^k_c$ is a $\sum_{v=1}^{V} d_v$-dimensional feature vector with $\boldsymbol{p}^k_c = \left[\boldsymbol{p}^{k,1}_c, \boldsymbol{p}^{k,2}_c, \ldots, \boldsymbol{p}^{k,V}_c\right]$. Similarly, another set of m_c clustering centers $\{\boldsymbol{n}^1_c, \boldsymbol{n}^2_c, \ldots, \boldsymbol{n}^{m_c}_c\}$ can be returned by invoking the spectral clustering procedure along with $\mathbf{W}_{\mathcal{I}^-_c}$. Following (Zhang and Wu 2015), the number of clustering centers m_c is set as:

$$m_c = \lceil \eta \cdot \min\left(|\mathcal{I}^+_c|, |\mathcal{I}^-_c|\right) \rceil \tag{8}$$

Here, the ratio parameter $\eta \in (0, 1)$ controls the number of clustering centers for embedding feature generation.

For any instance $\boldsymbol{x} = [\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^V] \in \mathcal{X}$, a total of $2m_c$ embedded features can be generated by querying the distance between $\boldsymbol{x}$ and the clustering centers on the v-th view:

$$\begin{aligned}\phi^v_c(\boldsymbol{x}) = \Big[&d(\boldsymbol{x}^v, \boldsymbol{p}^{1,v}_c), d(\boldsymbol{x}^v, \boldsymbol{p}^{2,v}_c), \ldots, d(\boldsymbol{x}^v, \boldsymbol{p}^{m_c,v}_c),\\ &d(\boldsymbol{x}^v, \boldsymbol{n}^{1,v}_c), d(\boldsymbol{x}^v, \boldsymbol{n}^{2,v}_c), \ldots, d(\boldsymbol{x}^v, \boldsymbol{n}^{m_c,v}_c)\Big]\end{aligned} \tag{9}$$

Here, $d(\cdot, \cdot)$ corresponds to the Euclidean distance. Therefore, the mapping function $\Phi_c : \mathcal{X} \to \mathcal{Z}_c$ will embed $\boldsymbol{x}$ into the $2m_c \cdot V$ feature space:

$$\Phi_c(\boldsymbol{x}) = [\phi^1_c(\boldsymbol{x}), \phi^2_c(\boldsymbol{x}), \ldots, \phi^V_c(\boldsymbol{x})] \tag{10}$$

To determine the relevancy of y_c, one binary training set $\widehat{D}_c$ is derived from $\widehat{D}$ based on Φ_c:

$$\widehat{D}_c = \{(\Phi_c(\boldsymbol{x}_i), \widehat{Y}_i(c)) \mid 1 \le i \le m\} \tag{11}$$

$$\text{where } \widehat{Y}_i(c) = \begin{cases} +1, & \text{if } y_c \in \widehat{Y}_i \\ -1, & \text{otherwise} \end{cases}$$

Accordingly, one binary classifier $f_c : \mathcal{Z}_c \to \mathbb{R}$ is induced by invoking binary training algorithm $\mathcal{B}$ on $\widehat{D}_c$, i.e. $f_c \leftarrow\!\shortmid \mathcal{B}(\widehat{D}_c)$. Given the unseen instance $\boldsymbol{u} \in \mathcal{X}$, its relevant label set Y is predicted as:

$$Y = \{y_c \mid f_c(\Phi_c(\boldsymbol{u})) \ge 0,\ 1 \le c \le q\} \tag{12}$$

Table 1 summarizes the complete procedure of GRADIS. Firstly, the weighted graph over all training examples is formed by fusing kNN-based similarity graphs over all views (Steps 1-4). After that, candidate label set of each training example is disambiguated via iterative label propagation over the weighted directed graph (Steps 5-7). Based on the disambiguated training examples, the predictive model is induced based on embedding features generated by exploiting clustering structure (Steps 8-13). Finally, the label set for unseen instance is predicted by querying the induced model (Step 14).

Table 2: Characteristic of experimental data sets. Here, p controls the fraction of examples which are partially labeled, and r controls the number of false positive labels which reside in the candidate label set.

Data Set	$\lvert\mathcal{S}\rvert$	$V(\mathcal{S})$	$Dim(\mathcal{S})$	$CL(\mathcal{S})$	$LCard(\mathcal{S})$	Domain	Controlling Parameters
Emotions	593	2	8 / 64	6	1.869	*music*	
Yeast	2,417	2	24 / 79	14	4.237	*biology*	
Corel5k	4,999	4	100 / 512 / 1,000 / 4,096	260	3.397	*image*	
Pascal	9,963	5	100 / 512 / 1,000 / 4,096 / 804	20	1.465	*image*	$p \in \{0.3, 0.5, 0.7\}$
Mirflickr	25,000	5	100 / 512 / 1,000 / 4,096 / 457	38	4.716	*image*	$r = 3$
Youku25k	24,940	4	64 / 128 / 2,048 / 2,048	114	2.130	*video*	
Youku50k	49,940	4	64 / 128 / 2,048 / 2,048	114	1.564	*video*	

Table 3: Experimental results of each comparing approach in terms of *ranking loss*, where the best performance (the smaller the better) on each data set and specific value of p is shown in bold face.

Data Set	p	GRADIS	GRADIS-B	ML-KNN	LIFT	LSAMML	F2L21F
Emotion	0.3	**0.145±0.021**	0.146±0.023	0.190±0.034	0.164±0.028	0.207±0.025	0.183±0.021
	0.5	**0.159±0.024**	0.173±0.030	0.217±0.029	0.188±0.046	0.215±0.037	0.201±0.045
	0.7	**0.191±0.028**	0.236±0.026	0.265±0.033	0.240±0.026	0.236±0.029	0.230±0.029
Yeast	0.3	0.173±0.011	**0.165±0.012**	0.174±0.009	0.178±0.012	0.302±0.018	0.345±0.014
	0.5	**0.167±0.011**	0.187±0.012	0.175±0.009	0.184±0.014	0.302±0.021	0.351±0.013
	0.7	**0.171±0.011**	0.196±0.012	0.179±0.009	0.186±0.010	0.303±0.020	0.349±0.013
Pascal	0.3	**0.122±0.006**	0.123±0.009	0.267±0.002	0.134±0.005	0.251±0.006	0.235±0.003
	0.5	**0.130±0.005**	0.139±0.006	0.272±0.006	0.148±0.008	0.254±0.007	0.249±0.005
	0.7	**0.143±0.007**	0.148±0.011	0.273±0.006	0.157±0.005	0.258±0.004	0.271±0.010
Corel5k	0.3	0.093±0.006	**0.089±0.006**	0.132±0.004	0.091±0.006	0.143±0.008	0.333±0.007
	0.5	0.098±0.006	0.095±0.007	0.135±0.006	**0.094±0.006**	0.147±0.007	0.339±0.007
	0.7	0.108±0.006	0.103±0.007	0.139±0.008	**0.097±0.004**	0.151±0.008	0.347±0.007
Mirflickr	0.3	**0.088±0.013**	0.091±0.015	0.170±0.012	0.108±0.009	0.174±0.008	0.137±0.011
	0.5	**0.092±0.007**	0.097±0.014	0.173±0.011	0.128±0.009	0.177±0.008	0.145±0.013
	0.7	**0.096±0.009**	0.103±0.015	0.174±0.011	0.147±0.011	0.178±0.010	0.148±0.010
Youku25k	0.3	**0.045±0.007**	0.068±0.011	0.130±0.011	0.074±0.013	0.078±0.008	0.045±0.011
	0.5	**0.049±0.007**	0.078±0.009	0.133±0.012	0.087±0.011	0.084±0.009	0.050±0.009
	0.7	**0.056±0.008**	0.089±0.009	0.138±0.010	0.101±0.011	0.093±0.008	0.057±0.009
Youku50k	0.3	**0.029±0.007**	0.051±0.013	0.117±0.009	0.066±0.010	0.067±0.010	0.035±0.013
	0.5	**0.033±0.010**	0.057±0.011	0.121±0.013	0.075±0.011	0.078±0.011	0.042±0.010
	0.7	**0.040±0.009**	0.066±0.011	0.129±0.010	0.092±0.010	0.085±0.009	0.049±0.008

Experiments

Experimental Setup

Data Sets To thoroughly evaluate the performance of comparing approaches, seven benchmark data sets are collected for experimental studies including `emotions` (Trohidis et al. 2008), `yeast` (Elisseeff and Weston 2002), `Corel5k` (Duygulu et al. 2002), `Pascal` (Everingham et al. 2010), `Mirflickr` (Huiskes and Lew 2008), `Youku25k` and `Youku50k` (Wu et al. 2019). Table 2 summarizes characteristics of each benchmark data set $\mathcal{S}$, including the # examples ($|\mathcal{S}|$), # views ($V(\mathcal{S})$), dimensionality of each view ($Dim(\mathcal{S})$), # class labels ($CL(\mathcal{S})$), and average # ground-truth labels per example (i.e. label cardinality $LCard(\mathcal{S})$).

Following the widely-used protocal for introducing labeling noise (Cour, Sapp, and Taskar 2011; Liu and Dietterich 2012; Tang and Zhang 2017; Yu and Zhang 2017; Chen, Patel, and Chellappa 2018), two controlling parameters p and r are utilized to generate MVPML examples with candidate label set from MVML examples. Here, p controls the fraction of examples in the data set which are partially labeled, i.e. with false positive labels in the associated label set. Furthermore, r controls the number of false positive labels which reside in the candidate label set. For an MVML example $(\boldsymbol{x}, \widetilde{Y})$ with ground-truth label set $\widetilde{Y}$, an MVPML example $(\boldsymbol{x}, Y)$ is generated by randomly adding r false positive labels $\Delta_r \subseteq \mathcal{Y} \setminus \widetilde{Y}$ into $\widetilde{Y}$, i.e. $Y = \widetilde{Y} \bigcup \Delta_r$.

In this paper, three configurations of p are considered with $p \in \{0.3, 0.5, 0.7\}$. Furthermore, the number of false positive labels r is set to be 3, which is comparable to the label cardinality of the experimental data sets shown in Table 2.

Comparing Approaches The problem of MVPML is firstly studied in this paper, where no existing MVPML are readily available for comparative studies.

As the MVPML problem degenerates to the MVML problem by treating all candidate labels as ground-truth ones, two state-of-the-art MVML approaches F2L21F (Zhu, Li, and Zhang 2016) and LSAMML (Zhang et al. 2018) are employed for performance comparison. Furthermore, other two well-established multi-label learning approaches ML-KNN (Zhang and Zhou 2007) and LIFT (Zhang and Wu 2015) are also employed as comparing approaches by concatenating

Table 4: Experimental results of each comparing approach in terms of *average precision*, where the best performance (the larger the better) on each data set and specific value of p is shown in bold face.

Data Set	p	GRADIS	GRADIS-B	ML-KNN	LIFT	LSAMML	F2L21F
Emotion	0.3	**0.817±0.030**	0.811±0.033	0.775±0.031	0.796±0.028	0.762±0.031	0.779±0.026
	0.5	**0.801±0.033**	0.787±0.029	0.760±0.024	0.777±0.048	0.749±0.042	0.759±0.052
	0.7	**0.772±0.026**	0.735±0.026	0.711±0.041	0.734±0.023	0.725±0.029	0.733±0.034
Yeast	0.3	0.763±0.011	**0.765±0.013**	0.757±0.012	0.756±0.018	0.432±0.016	0.606±0.016
	0.5	**0.761±0.015**	0.755±0.011	0.756±0.011	0.751±0.019	0.442±0.022	0.601±0.013
	0.7	**0.757±0.014**	0.750±0.013	0.752±0.012	0.748±0.013	0.423±0.019	0.601±0.015
Pascal	0.3	**0.684±0.014**	0.680±0.016	0.456±0.006	0.638±0.013	0.471±0.009	0.529±0.006
	0.5	**0.667±0.012**	0.666±0.011	0.451±0.006	0.590±0.040	0.463±0.008	0.496±0.013
	0.7	**0.640±0.010**	0.654±0.014	0.450±0.004	0.597±0.009	0.461±0.004	0.461±0.011
Corel5k	0.3	0.473±0.011	**0.483±0.011**	0.344±0.006	0.467±0.011	0.358±0.012	0.210±0.007
	0.5	0.464±0.010	**0.472±0.010**	0.343±0.007	0.459±0.013	0.356±0.011	0.198±0.008
	0.7	0.453±0.011	**0.464±0.012**	0.340±0.006	0.446±0.012	0.354±0.011	0.186±0.008
Mirflickr	0.3	**0.750±0.013**	0.749±0.009	0.509±0.011	0.656±0.008	0.533±0.010	0.656±0.011
	0.5	**0.739±0.009**	0.735±0.013	0.505±0.014	0.623±0.009	0.529±0.008	0.644±0.014
	0.7	**0.730±0.011**	0.720±0.015	0.505±0.015	0.474±0.013	0.529±0.010	0.634±0.009
Youku25k	0.3	**0.680±0.008**	0.657±0.09	0.473±0.008	0.550±0.009	0.551±0.010	0.654±0.009
	0.5	**0.675±0.009**	0.633±0.010	0.468±0.010	0.504±0.009	0.546±0.010	0.646±0.009
	0.7	**0.664±0.008**	0.618±0.010	0.460±0.010	0.474±0.011	0.539±0.011	0.633±0.008
Youku50k	0.3	**0.701±0.010**	0.679±0.009	0.488±0.009	0.568±0.009	0.565±0.013	0.677±0.011
	0.5	**0.696±0.009**	0.658±0.010	0.476±0.014	0.517±0.009	0.556±0.010	0.665±0.009
	0.7	**0.688±0.012**	0.633±0.012	0.470±0.010	0.488±0.012	0.549±0.011	0.652±0.009

the multi-view representation into a single one. A simplified variant of GRADIS (named as GRADIS-B) is also evaluated to show the usefulness of candidate label set disambiguation. which works in the same way as GRADIS except that the disambiguation stage is ignored from the training procedure.

For the comparing approaches, parameters suggested in respective literatures (Zhang and Zhou 2007; Zhang and Wu 2015; Zhu, Li, and Zhang 2016; Zhang et al. 2018) are used for experimental studies. As shown in Table 1, parameters of GRADIS are set as $k = 8$, $\alpha = 0.95$, $\gamma = 0.1$ and $\eta = 0.1$. For performance evaluation, six popular multi-label metrics including *hamming loss*, *ranking loss*, *one-error*, *coverage*, *average precision* and *macro-averaging AUC* (Zhang and Zhou 2014) are utilized in this paper. For the first four metrics, the smaller the metric value the better the performance. For the other two metrics, the larger the metric value the better the performance.[3] Ten-fold cross-validation is performed on each data set, where the mean metric value as well as standard deviation are recorded.

Experimental Results

Comparative Studies Tables 3 to 5 report the detailed experimental results of each comparing approach in terms of *ranking loss*, *average precision* and *macro-averaging AUC*. On each data set, the best performance of all comparing approaches w.r.t. specific value of p is shown in boldface.[4]

Furthermore, Friedman test (Demšar 2006) is employed for statistical performance comparison among the comparing approaches. At significance level 0.05, the null hypothesis of equal performance among all comparing approaches is rejected in terms of each evaluation metric. Consequently, *Bonferroni-Dunn test* (Demšar 2006) is employed as the post-hoc test to show the relative performance among comparing approaches by treating GRADIS as the control approach.

Figure 2 illustrates the critical difference (CD) diagrams where the average rank of each approach is marked along the axis with lower ranks to the right. Any approach whose average rank is within one CD with GRADIS is interconnected to each other with a thick line. Otherwise, it is regarded to have significantly different performance against GRADIS.

Based on the reported experimental results, it is impressive to observe that:

- Out of 126 statistical tests (7 data sets $\times$ 3 configurations of controlling parameter p $\times$ 6 evaluation metrics), GRADIS achieves better metric value than ML-KNN, LIFT, LSAMML and F2L21F in 96.82%, 88.89%, 97.62% and 88.10% cases. Furthermore, the simplified variant GRADIS-B also achieves better metric value than ML-KNN, LIFT, LSAMML and F2L21F in 94.44%, 81.75%, 91.27% and 76.98% cases.
- As shown in Figure 2, GRADIS achieves lowest (best) average rank in terms of all evaluation metrics. In addition, GRADIS significantly outperforms ML-KNN and LSAMML in terms of all evaluation metrics, and significantly outperforms LIFT and F2L21F in terms of *hamming loss*, *ranking loss*, *one-error* and *average precision*.

[3] Due to page limit, detailed definitions on the evaluation metrics can be found in (Zhang and Zhou 2014; Gibaja and Ventura 2015)

[4] Detailed experimental results in terms of the other evaluation metrics are not reported here due to page limit while similar observations can be made as well.

Table 5: Experimental results of each comparing approach in terms of *macro-averaging AUC*, where the best performance (the larger the better) on each data set and specific value of p is shown in bold face.

Data Set	p	GRADIS	GRADIS-B	ML-KNN	LIFT	LSAMML	F2L21F
Emotion	0.3	**0.850±0.022**	0.843±0.022	0.786±0.020	0.825±0.028	0.792±0.027	0.809±0.026
	0.5	**0.834±0.023**	0.812±0.031	0.754±0.024	0.794±0.040	0.780±0.037	0.788±0.032
	0.7	**0.805±0.040**	0.747±0.044	0.705±0.039	0.749±0.024	0.744±0.037	0.746±0.039
Yeast	0.3	0.666±0.012	0.672±0.019	0.669±0.018	0.660±0.018	0.495±0.022	**0.674±0.016**
	0.5	0.667±0.018	0.666±0.025	0.654±0.010	0.669±0.012	0.502±0.023	**0.672±0.011**
	0.7	**0.659±0.019**	0.651±0.022	0.641±0.017	0.657±0.017	0.491±0.027	0.658±0.014
Pascal	0.3	0.832±0.015	**0.834±0.011**	0.601±0.014	0.828±0.005	0.658±0.007	0.707±0.011
	0.5	**0.805±0.011**	0.801±0.013	0.581±0.004	0.812±0.008	0.655±0.005	0.694±0.002
	0.7	**0.793±0.011**	0.788±0.013	0.585±0.010	0.794±0.008	0.645±0.005	0.673±0.011
Corel5k	0.3	**0.814±0.007**	0.802±0.014	0.650±0.007	0.811±0.004	0.738±0.011	0.601±0.012
	0.5	**0.797±0.012**	0.781±0.015	0.646±0.010	0.797±0.015	0.731±0.011	0.592±0.012
	0.7	**0.767±0.014**	0.762±0.010	0.639±0.010	0.793±0.008	0.722±0.010	0.592±0.013
Mirflickr	0.3	0.657±0.011	0.657±0.015	0.533±0.016	**0.660±0.020**	0.599±0.019	0.656±0.013
	0.5	**0.655±0.014**	0.650±0.022	0.520±0.023	0.652±0.019	0.587±0.013	0.647±0.010
	0.7	**0.646±0.019**	0.644±0.014	0.509±0.018	0.642±0.017	0.569±0.013	0.639±0.009
Youku25k	0.3	0.900±0.009	0.892±0.010	0.774±0.010	0.891±0.013	0.900±0.009	**0.946±0.016**
	0.5	0.888±0.013	0.879±0.012	0.770±0.013	0.886±0.009	0.886±0.014	**0.935±0.017**
	0.7	0.874±0.009	0.869±0.009	0.765±0.015	0.879±0.014	0.872±0.011	**0.924±0.015**
Youku50k	0.3	0.905±0.013	0.900±0.009	0.781±0.009	0.897±0.014	0.905±0.010	**0.949±0.010**
	0.5	0.892±0.010	0.887±0.012	0.776±0.013	0.890±0.011	0.890±0.009	**0.940±0.016**
	0.7	0.883±0.009	0.878±0.015	0.769±0.010	0.883±0.015	0.878±0.013	**0.927±0.012**

- As shown in Tables 3 to 5, the performance gap between GRADIS and GRADIS-B becomes more pronounced as the level of labeling noise (i.e. p) increases. Furthermore, the average rank of GRADIS is smaller than GRADIS-B in terms of all evaluation metrics. These results validate the usefulness of the disambiguation stage employed by GRADIS, especically when the level of labeling noise is high.

Sensitivity Analysis As shown in Table 1, the implementation of GRADIS involves four parameters k (# nereast neighbors), α (balancing parameter), γ (thresholding parameter), and η (ratio parameter). To investigate the parameter sensitivity of GRADIS, Figure 3 gives an illustrative example of how the performance of GRADIS (in terms of *average precision*) changes as the value of each parameter varies on the `Yeast` data set. Here, when the value of one parameter varies, the values for the other parameters are fixed as $k = 8$, $\alpha = 0.95$, $\gamma = 0.1$ and $\eta = 0.1$.

As shown in Figure 3, it is shown that in most cases: a) The performance of GRADIS is relatively stable when k takes value in [5,11]; b) The performance of GRADIS is relatively stable when α takes value in [0.7,0.95]; c) The performance of GRADIS decreases as γ exceeds 0.1; d) The performance of GRADIS decreases as η is relatively stable when η takes value in [0.05,0.35]. In light of these observations, we have used the parameter configuration $k = 8$, $\alpha = 0.95$, $\gamma = 0.1$ and $\eta = 0.1$ for GRADIS in experimental studies.

Related Work

In this paper, the problem of multi-view partial multi-label learning is firstly studied, which is closely-related to multi-view multi-label learning as well as partial label learning.

Multi-view multi-label learning (MVML) deals with the problem where each example is represented by multiple views while associated with multiple class labels. The basic assumption of MVML lies in that each of the associated labels is valid in characterizing semantics of the example. To learn from MVML examples, most works focus on exploiting shared subspace to enable information communication and fusing from different views. In (Liu et al. 2015), a low-dimensional shared representation is learned by enforcing the low-rank constraint which is suitable for image classification based on matrix completion. In (Zhu, Li, and Zhang 2016), block-wise regularization is introduced to remove redundancy views and noisy features. In (Zhan and Zhang 2017; Xing et al. 2018), the popular co-training framework is adapted for multi-label learning by considering the reliability of labeling information to be communicated among different views. In (Zhang et al. 2018; Zhu et al. 2018), the shared subspace is learned by exploiting multi-view correlations via Hilbert-Schmidt Independence Criterion (HSIC) or matrix factorization. In (Wu et al. 2019), the shared subspace as well as view-specific information extraction are jointly utilized for model induction.

Partial label learning (PLL) deals with the problem where each example is represented by a single instance while associated with multiple *candidate* labels. The basic assumption of PLL lies in that all the associated labels are only candidate ones among which only one is valid. To learn from PLL examples, most works focus on disambiguating the candidate label set for model induction. For identification-based disambiguation, the ground-truth label is treated as latent variable whose value is esitimated via iterative procedure such as EM (Liu and Dietterich 2012; Yu and Zhang 2017; Chen, Patel, and Chellappa 2018).

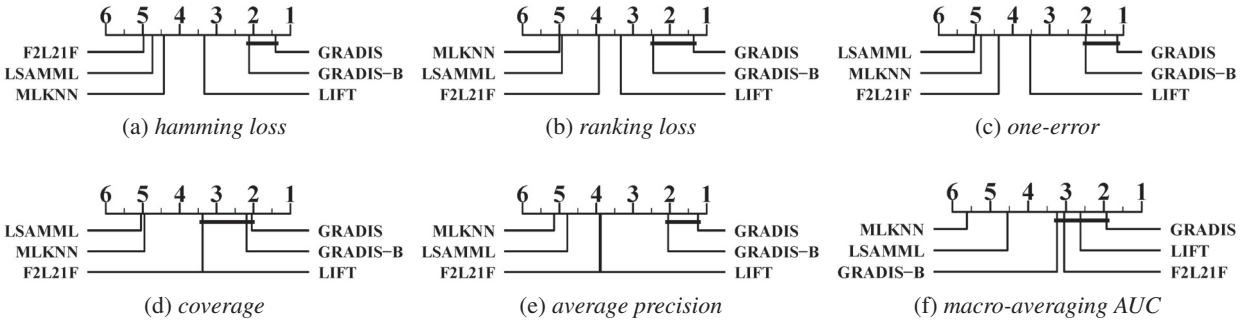
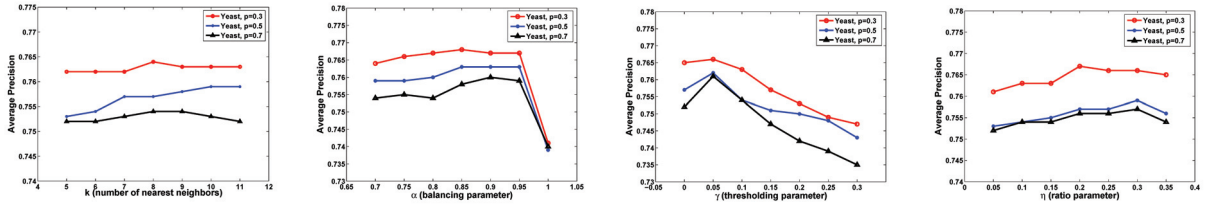


Figure 2: Comparison of GRADIS (control approach) against other approaches with *Bonferroni-Dunn* test. Approaches not connected with GRADIS in the CD diagram are considered to have significantly different performance from the control approach (CD=1.7597).



(a) *Varying* k (b) *Varying* α (c) *Varying* γ (d) *Varying* η

Figure 3: The performance of GRADIS (in terms of *average precision*) changes as the value of each parameter varies. (a) k (# nearest neighbors) increases from 5 to 11 with step-size 1; (b) α (balancing parameter) increases from 0.7 to 1 with step-size 0.05; (c) γ (thresholding parameter) increases from 0 to 0.3 with step-size 0.05; (d) η (ratio parameter) increases from 0.05 to 0.35 with step-size 0.05.

For averaging-based disambiguation, each candidate label is treated in an equal manner whose modeling outputs are averaged for final prediction (Cour, Sapp, and Taskar 2011; Tang and Zhang 2017). Recently, studies on partial label learning which assume non-unique ground-truth label in candidate label set (Xie and Huang 2018; Yu et al. 2018; Fang and Zhang 2019; Sun et al. 2019) have also been investigated.

Conclusion

In MVPML, the multi-view training example is assumed to have multiple labeling assignments which are only partially valid. In this paper, a two-stage approach towards MVPML is proposed via graph-based disambiguation. Firstly, similarity graph from multiple views are fused to enable candidate label set disambiguation via iterative label propagation. After that, disambiguation-guided clustering analysis is performed to generate embedded features for training the predictive model. Experimental studies over a number of benchmark data sets show that the proposed approach serves as an effective solution to learn from MVPML examples.

Acknowledgments

This work was completed while Ze-Sen Chen was conducting his internship at the YouKu Cognitive and Intelligent Lab, Alibaba Group. The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Key R&D Program of China (2018YFB1004300), the National Science Foundation of China (61573104), and partially supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Chen, C.-H.; Patel, V. M.; and Chellappa, R. 2018. Learning from ambiguously labeled face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(7):1653–1667.

Cour, T.; Sapp, B.; and Taskar, B. 2011. Learning from partial labels. *Journal of Machine Learning Research* 12(May):1501–1536.

Demšar, J. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7(Jan):1–30.

Duygulu, P.; Barnard, K.; de Freitas, J. F.; and Forsyth, D. A. 2002. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the 7th European Conference on Computer Vision*, 97–112.

Elisseeff, A., and Weston, J. 2002. A kernel method for multi-labelled classification. In Dietterich, T. G.; Becker, S.;

and Ghahramani, Z., eds., *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press. 681–687.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision* 88(2):303–338.

Fang, J.-P., and Zhang, M.-L. 2019. Partial multi-label learning via credible label elicitation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 3518–3525.

Gibaja, E., and Ventura, S. 2015. A tutorial on multilabel learning. *ACM Computing Surveys* 47(3):Article 52.

Huang, J.; Li, G.; Huang, Q.; and Wu, X. 2016. Learning label-specific features and class-dependent labels for multi-label classification. *IEEE Transactions on Knowledge and Data Engineering* 28(12):3309–3323.

Huiskes, M. J., and Lew, M. S. 2008. The MIR Flickr retrieval evaluation. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, 39–43.

Liu, L., and Dietterich, T. 2012. A conditional multinomial mixture model for superset label learning. In Bartlett, P.; Pereira, F. C. N.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 25*. Cambridge, MA: MIT Press. 557–565.

Liu, M.; Luo, Y.; Tao, D.; Xu, C.; and Wen, Y. 2015. Low-rank multi-view learning in matrix completion for multi-label image classification. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2778–2784.

Luo, Y.; Tao, D.; Xu, C.; Xu, C.; Liu, H.; and Wen, Y. 2013. Multiview vector-valued manifold regularization for multilabel image classification. *IEEE Transactions on Neural Networks and Learning Systems* 24(5):709–722.

Sun, L.; Feng, S.; Wang, T.; Lang, C.; and Jin, Y. 2019. Partial multi-label learning by low-rank and sparse decomposition. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, in press.

Tang, C.-Z., and Zhang, M.-L. 2017. Confidence-rated discriminative partial label learning. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2611–2617.

Trohidis, K.; Tsoumakas, G.; Kalliris, G.; and Vlahavas, I. 2008. Multilabel classification of music into emotions. In *Proceedings of the 2008 International Conference on Music Information Retrieval*, 325–330.

von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17(4):395–416.

Weng, W.; Lin, Y.; Wu, S.; Li, Y.; and Kang, Y. 2018. Multi-label learning based on label-specific features and local pairwise label correlation. *Neurocomputing* 273:385–394.

Wu, X.; Chen, Q.-G.; Hu, Y.; Wang, D.; Chang, X.; Wang, X.; and Zhang, M.-L. 2019. Multi-view multi-label learning with view-specific information extraction. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 3884–3890.

Xie, M.-K., and Huang, S.-J. 2018. Partial multi-label learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 4302–4309.

Xing, Y.; Yu, G.; Domeniconi, C.; Wang, J.; and Zhang, Z. 2018. Multi-label co-training. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2882–2888.

Yu, F., and Zhang, M.-L. 2017. Maximum margin partial label learning. *Machine Learning* 106(4):573–593.

Yu, G.; Chen, X.; Domeniconi, C.; Wang, J.; Li, Z.; Zhang, Z.; and Wu, X. 2018. Feature-induced partial multi-label learning. In *Proceedings of the 18th IEEE International Conference on Data Mining*, 1398–1403.

Zhan, W., and Zhang, M.-L. 2017. Inductive semi-supervised multi-label learning with co-training. In *Proceedings of the 23rd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1305–1314.

Zhang, M.-L., and Wu, L. 2015. LIFT: Multi-label learning with label-specific features. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(1):107–120.

Zhang, M. L., and Zhou, Z. H. 2007. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition* 40(7):2038–2048.

Zhang, M.-L., and Zhou, Z.-H. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26(8):1819–1837.

Zhang, C.; Yu, Z.; Hu, Q.; Zhu, P.; Liu, X.; and Wang, X. 2018. Latent semantic aware multi-view multi-label classification. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 4414–4421.

Zhu, P.; Hu, Q.; Hu, Q.; Zhang, C.; and Feng, Z. 2018. Multi-view label embedding. *Pattern Recognition* 84:126–135.

Zhu, X.; Li, X.; and Zhang, S. 2016. Block-row sparse multiview multilabel learning for image classification. *IEEE Transactions on Cybernetics* 46(2):450–461.