Can Label-Specific Features Help Partial-Label Learning?

Ruo-Jing Dong, Jun-Yi Hang, Tong Wei[*], Min-Ling Zhang

School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
drj2364082@163.com, {hangjy, weit, zhangml}@seu.edu.cn

Abstract

Partial label learning (PLL) aims to learn from *inexact* data annotations where each training example is associated with a coarse candidate label set. Due to its practicability, many PLL algorithms have been proposed in recent literature. Most prior PLL works attempt to identify the ground-truth labels from candidate sets and the classifier is trained afterward by fitting the features of examples and their exact ground-truth labels. From a different perspective, we propose to enrich the feature space and raise the question "*Can label-specific features help PLL?*" rather than learning from examples with identical features for all classes. Despite its benefits, previous label-specific features approaches rely on ground-truth labels to split positive and negative examples of each class and then conduct clustering analysis, which is not directly applicable in PLL. To remedy this problem, we propose an uncertainty-aware confidence region to accommodate false positive labels. We first employ graph-based label enhancement to yield smooth pseudo-labels and facilitate the confidence region split. After acquiring label-specific features, a family of binary classifiers is induced. Extensive experiments on both synthesized and real-world datasets are conducted and the results show that our method consistently outperforms eight baselines. Our code is released at https://github.com/meteoseeker/UCL

Introduction

Partial label learning (PLL) is an important problem which has been studied a lot in the past decade (Cour, Sapp, and Taskar 2011; Chen, Patel, and Chellappa 2017; Feng et al. 2020; Lv et al. 2020; Wen et al. 2021; Wang et al. 2022). PLL aims to learn a classifier from training datasets with *inexact* labels, i.e., each instance is associated with a set of candidate labels among which only one is correct (Cour, Sapp, and Taskar 2011; Lyu et al. 2020; Bao, Hang, and Zhang 2021). Notice that, PLL is also known as superset label learning (Liu and Dietterich 2012; Gong et al. 2017) or ambiguous label learning (Hüllermeier and Beringer 2006; Chen et al. 2014). Unlike ordinary multi-class classification problems where each training instance is annotated with one ground-truth label, PLL aims to train classifiers from ambiguously (inexact) candidate labels directly (Zhou 2018), thus reducing the cost of data annotation. Confronting the expensive cost of manual labeling, PLL has been extensively studied and adopted for various real-world applications, e.g., image annotation (Zeng et al. 2013), web mining (Luo and Orabona 2010), ecoinformatics (Liu and Dietterich 2012), and natural language processing (Zhou et al. 2018).

The key of PLL is to identify the ground-truth labels from candidate label sets and many effective approaches have been proposed. Existing approaches work by manipulating label space including the average-based strategy (Hüllermeier and Beringer 2006; Cour, Sapp, and Taskar 2011) and identification-based strategy (Nguyen and Caruana 2008; Liu and Dietterich 2012; Yu and Zhang 2016). Some other works explore the feature space (Zhang, Zhou, and Liu 2016; Bao, Hang, and Zhang 2021; Wang, Zhang, and Li 2021) to make use of potentially useful structural information. However, none of them consider the exploration of specific semantic relations between instances and labels, rather than leveraging identical features for all classes.

Therefore, this paper attempts to deal with PLL from a different perspective, i.e., the label-specific features, which is an effective feature enhancement approach. In particular, we raise the question that "Can label-specific features help PLL?". Originally, the label-specific features are proposed in multi-label learning(Zhang and Zhou 2013; Zhang and Wu 2014; Huang et al. 2015; Yu and Zhang 2021; Wei, Tu, and Li 2019). It first generates a specific feature description for each class and the classifier is then trained by fitting the label-specific features and their corresponding labels. By exploiting label-specific features, it achieves superior classification performance for its benefits of class discrimination features. For example, in automatic image annotation, texture-based features would be preferred in discriminating silk and non-silk images, and color-based features would be preferred in discriminating ocean and non-ocean images. For another example, in automatic text categorization, features associated with terms like "carnivores", "predator", and "habitat" would be informative in discriminating ecological and non-ecological documents while features related to terms like "petroleum", "mineral", and "diastrophism" would be informative in discriminating geological and non-geological documents.

Despite the benefits of the label-specific features, it cannot be directly adopted to PLL because conventional label-

[*]Corresponding author

specific features approaches rely on *exact* data annotations. Specifically, for each class, they first partition the data into positive and negative examples by the ground-truth labels, then conduct the clustering analysis to obtain cluster centers. The label-specific features are then calculated based on the distance between samples and clusters. However, the ground-truth labels are not accessible in PLL, which is the key obstacle to generating label-specific features.

To overcome this challenge, this paper constructs the *UnCertainty-aware Label-specific features* (UCL) that explores underlying true labels of training samples and incorporates the uncertainty to improve the label-specific features generation procedure. First, we employ a graph-based label enhancement to refine the given candidate labels, which can exploit the structural information of data. Then, for each class, the data can be split into three parts, i.e., *positive*, *uncertain*, and *negative* samples. This is realized based on the output pseudo-labeling confidence by the graph-based label enhancement module. The introduced *uncertain* region can incorporate false positive labels in the candidates. We conduct spectral clustering to each part of the examples and collect cluster centers as their representatives. After that, the label-specific features are obtained by computing the distances between examples and clusters. Finally, a one-vs-rest classifier is learned using the newly formed training dataset with label-specific features. Extensive experiments on synthesized and real-world datasets demonstrate the superiority of the proposed approach and the effectiveness of label-specific features. To sum up, our contributions are:

1. We for the first time study the label-specific features in PLL which has the potential to inspire more research on exploiting the feature space enhancement;
2. We extend the supervised label-specific features to PLL by proposing an uncertainty-aware confidence thresholding approach to deal with inexact annotations;
3. Our method achieves superior performance against eight existing PLL algorithms on ten datasets (five synthetic and five real-world datasets).
4. We study the case where the effect of label-specific features is not noticeable, which can guide the future design of the learning framework.

Related Work

Partial Label Learning Most existing PLL algorithms attempt to learn from ambiguous data annotations via candidate label disambiguation, which is conducted in the label space. Corresponding strategies can be roughly divided into two types. One intuitive strategy is the averaging-based method, which simply treats each candidate label of an instance equally for training. For instance, some approaches (Cour, Sapp, and Taskar 2011; Tang and Zhang 2017) attempt to distinguish the averaged assignments for all candidate labels from the assignments of non-candidate labels. Some other approaches (Hüllermeier and Beringer 2006; Zhang and Yu 2015; Gong et al. 2017) classify the unseen instance by voting the candidate labels of its neighbors. Despite the simplicity of the averaging-based strategy, one potential defect lies in that the learning process is usually affected by false positive labels because it does not filter out incorrect candidate labels.

Another strategy of candidate label disambiguation is to identify the ground-truth labels which are treated as latent variables during the model training phase. Generally, iterative optimization procedure such as EM is utilized to predict the latent variables, which can be achieved by likelihood maximum (Jin and Ghahramani 2002; Liu and Dietterich 2012) or margin maximum approaches (Nguyen and Caruana 2008; Yu and Zhang 2016; Chai, Tsang, and Chen 2019). However, a potential drawback of this strategy is that the identified label may turn out to be a false positive rather than the ground-truth label, which hurts the performance.

Apart from the above strategies, some recent works propose to manipulate the feature space. These methods attempt to dig out the ignored information in feature space and make full use of it, which is flexible to conduct in practice. For instance, a few approaches (Zhang, Zhou, and Liu 2016; Wang, Zhang, and Li 2021) generate latent labeling confidence over candidate label set by utilizing the graph structure among training data. Some approaches (Wu and Zhang 2019; Bao, Hang, and Zhang 2021) conduct dimensionality reductions on training data by learning a projection matrix.

Label-Specific Features The key idea of label-specific features is to generate different instance representations for each class. The methods of label-specific features generation can be roughly divided into two categories, i.e., feature selection and prototype-based feature transformation.

Feature selection methods generate label-specific features by retaining a specific subset from the original feature set for each class. Representative works such as LLSF use lasso regression (Huang et al. 2015) for label-specific features selection while considering label correlations. A subsequent work employs regularized optimization (Huang et al. 2017; Zhang et al. 2018) for feature selection, which first maps the features to a high-dimensional space and then selects the most prominent features (Kashef and Nezamabadi-pour 2019).

Label-specific features can also be generated by transforming prototypes of each class label. For example, the seminal work LIFT (Zhang and Wu 2014) provides a three-stage framework to perform prototype-based label-specific features transformation. To be specific, for each class, positive/negative prototypes are collected by performing the clustering analysis on its positive/negative training examples. Subsequently, label-specific features are generated via querying distances between examples and the gathered prototypes, which are utilized to induce a classification model. Extensive efforts have been made in this line of research to improve the multi-stage framework such as implementing feature reduction (Xu et al. 2016) or incorporating local pairwise label correlation (Weng et al. 2018). In addition, an end-to-end prototype-based approach is recently proposed (Hang et al. 2022) to reduce the defects brought by the decoupling nature of the previous multi-stage approaches.

Although label-specific features improve the performance in multi-label learning problems, it has not been studied in PLL yet. To fill this gap, this paper raises an intrinsic question "Can label-specific features help PLL?" and presents a new effective method to improve the performance.

The Proposed Approach

Problem Setup The goal of PLL is to learn a classifier $f : \mathcal{X} \rightarrow \mathcal{Y}$ using the training set $\mathcal{D} = (\mathbf{X}, \mathbf{Y})$. We denote $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \in \mathbb{R}^{N \times d}$ as the feature matrix and $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N] \in \{0, 1\}^{N \times L}$ as the corresponding label matrix, in which $y_{ij} = 1$ signifies that the j-th label is a candidate label of the i-th instance $\mathbf{x}_i$ whilst $y_{ij} = 0$ signifies the j-th label is a non-candidate label. We denote N and L as the number of training examples and classes, respectively.

Main Idea This paper explores the merit of label-specific features for PLL. The main obstacle is that the generation of label-specific features requires ground-truth labels in previous works (Zhang and Wu 2014; Wei, Tu, and Li 2019), which is not accessible in PLL. Therefore, we first introduce the *graph-based label enhancement* procedure to refine candidate labels; then an *uncertainty-aware label-specific features* generation approach is presented to deal with the uncertainty of being the ground-truth labels.

Graph-based Label Enhancement

We propose a label enhancement approach to propagate pseudo-labels on the undirected graph $G = \langle V, E \rangle$ by leveraging similarities between data. V and E denote the set of graph vertices and edges, respectively. In graph G, the similarities between vertices are encoded by a weight matrix $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_N]$. Considering the efficiency and scalability, we adopt the k-NN matrix to represent the similarity of each pair of data points $(\mathbf{x}_i, \mathbf{x}_j), 1 \le i, j \le N$ as follows:

$$w_{ij} := \begin{cases} \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), & \text{if } i \neq j \wedge \mathbf{x}_i \in \mathbf{NN}_k(\mathbf{x}_j) \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Here, we adopt the Gaussian kernel to measure the similarities and σ^2 is the bandwidth of the kernel, which is a popular choice in previous related works. Moreover, we employ the KD-Tree algorithm (Bentley 1975) to effectively calculate the k nearest neighbors denoted by $\mathbf{NN}_k$, which reduces the computational burden. To capture high-order graph information, prior works design models on the assumption that labels vary smoothly over the edges of the graph. To do so, we first normalize the weight matrix by $w_{ij} = w_{ij} / \sum_{k=1}^{N} w_{ik}$, and then propagate candidate labels for label disambiguation (Zhang and Yu 2015). Let $\tilde{\mathbf{Y}}$ denote the soft pseudo-label matrix, which is iteratively updated during the label propagation process. Specifically, at the $t-$ iteration, we have:

$$\tilde{\mathbf{Y}}^t = \beta \mathbf{W}^\top \tilde{\mathbf{Y}}^{t-1} + (1 - \beta)\mathbf{P} \tag{2}$$

Here, $\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_N]$ is the normalized partial label matrix, i.e., $p_{ij} = y_{ij} / \sum_{k=1}^{L} y_{ik}, \forall 1 \le i \le N$, parameter $\beta \in (0, 1)$ controls the balance between propagated pseudo-labels and initial labels. Noted that we set $\tilde{\mathbf{Y}}^0 = \mathbf{Y}$ as initialization. In Eq. (2), all nodes propagate candidate labels to their neighbors according to edge weights. α is used to trade off between information from neighborhoods and vertices themselves and we simply set it to 0.5 in all experiments. After each iteration, we normalize $\tilde{\mathbf{Y}}^t$ by:

$$\tilde{y}_{ij}^t = \frac{\tilde{y}_{ij}^t \cdot y_{ij}}{\sum_{k=1}^{L} \tilde{y}_{ik}^t \cdot y_{ik}}, \forall 1 \le i \le N \tag{3}$$

The label propagation process continues until convergence or reaches pre-defined maximum iterations. Based on the enhanced label matrix, we introduce the uncertainty-aware label-specific features approach in the next section.

Uncertainty-aware Label-specific Features

Unlike previous works which divide training instances into positive and negative categories, our approach partitions the training examples into positive, negative, and uncertain parts, which are respectively denoted by $\mathcal{P}$, $\mathcal{N}$, and $\mathcal{U}$. Specifically, for the j-th class, we have:

$$\begin{aligned} \mathcal{P}_j &= \{\mathbf{x}_i \mid (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}, \tilde{y}_{ij} > \tau_h\}, \\ \mathcal{N}_j &= \{\mathbf{x}_i \mid (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}, \tilde{y}_{ij} < \tau_l\}, \\ \mathcal{U}_j &= \{\mathbf{x}_i \mid (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}, \tau_l \le \tilde{y}_{ij} \le \tau_h\} \end{aligned} \tag{4}$$

Here, τ_h and τ_l are confidence thresholding parameters that are set as constants in our experiments.

The rationale behind Eq. (4) is that we leverage the confidence to represent the uncertainty of an example. For each class, examples associated with high confidence are classified as positive ones, which indicates the current label achieves an overwhelming victory against the other candidates, whilst examples associated with low confidence are categorized as negative ones, which usually are not related to the current label or in some cases have a large number of equivalent candidate labels. However, there exists a portion of examples difficult to classify, which are usually labeled with few highly competitive candidate labels. Examples uncertain about their observed labels are usually confusing and might have a misleading impact on feature generation if they participated in the following clustering tasks. Thus, we single out and deal with uncertain examples separately.

Moreover, PLL usually suffers from the class imbalance problem (Zhang and Zhou 2013), where the number of positive examples for each class is much less than the number of negative ones, i.e., $|\mathcal{P}_j| \ll |\mathcal{N}_j|$. Following the thought of LIFT, clustering information gained from positive examples as well as negative examples is treated with equal importance. By setting a ratio parameter r, we obtain the number of clusters of positive (m_j), negative (m_j), and uncertain (c_j) parts for the j-th class as follows:

$$\begin{aligned} m_j &= \lceil r \cdot \min\left(|\mathcal{P}_j|, |\mathcal{N}_j|\right) \rceil, \\ c_j &= \lceil r \cdot |\mathcal{U}_j| \rceil \end{aligned} \tag{5}$$

Notice that in a representative work LIFT, it chooses the k-means clustering to explore the local structures underlying positive and negative instances. However, in our approach we instead choose spectral clustering (Shi and Malik 2000; Von Luxburg 2007) inspired by spectral instance alignment (Zhang, Fang, and Li 2015), which considers the correlation between positive and negative instances. The goal of spectral clustering is to divide data (represented as a similarity graph) into several groups where the similarity between data points is high in the same group whilst low in different groups. After collecting the cluster centers for each part, we construct the label-specific features transformation $\phi_j : \mathcal{X} \rightarrow \mathcal{Z}_j$ for

Algorithm 1: The label-specific features approach in UCL

```
Inputs:
$\mathcal{D}$: the PL training set $\mathcal{D} = \{(\mathbf{X}, \mathbf{Y})\}$
$r$: the ratio parameter controlling the number of clusters
$k$: the number of nearest neighbors
$\tau_h, \tau_l$: the high/low partition thresholds
$\alpha$: the discount parameter
Output: feature transformation $\phi$
1: calculate similarity matrix $\mathbf{W}$ using Eq. (1)
2: refine the pseudo-label matrix $\tilde{\mathbf{Y}}$ using Eq. (2)
3: for $j = 1$ to $L$ do
4:     partition the training examples into positive, negative,
       and uncertain parts using Eq. (4)
5:     perform spectral clustering on three parts respectively
6:     set $\phi_j$ with the cluster centers using Eq. (6)
7: end for
8: return the label-specific features transformation $\phi_j$
```

the j-th class as follows:

$$\begin{aligned} \phi_j(\mathbf{x}) = \Big[& d\left(\mathbf{x}, \mathbf{p}_1^j\right), \cdots, d\left(\mathbf{x}, \mathbf{p}_{m_j}^j\right), \\ & d\left(\mathbf{x}, \mathbf{n}_1^j\right), \cdots, d\left(\mathbf{x}, \mathbf{n}_{m_j}^j\right), \\ & \alpha \times d\left(\mathbf{x}, \mathbf{u}_1^j\right), \cdots, \alpha \times d\left(\mathbf{x}, \mathbf{u}_{c_j}^j\right) \Big] \end{aligned} \tag{6}$$

Here, $\mathbf{p}_i^j, \mathbf{n}_i^j, \mathbf{u}_i^j$ are clustering centers for positive, negative, and uncertain parts, respectively. $d(\cdot,\cdot)$ is the distance measure and is set to Euclidean distance in our experiments. Function ϕ_j transforms the original d-dimensional input feature space to a $(2m_j + c_j)$-dimensional label-specific feature space. α is a discount parameter to put a large weight on positive and negative clusters. Algorithm 1 summarizes the pseudo-code of label-specific features generation procedure.

UCL can cooperate with many existing PLL classifiers such as PL-KNN (Hüllermeier and Beringer 2006), PL-SVM (Nguyen and Caruana 2008), and SURE (Feng and An 2019). In the experiments of this paper, we choose SURE as the default classifier for its superior performance.

Experiments

We conduct experiments on ten datasets (five synthetic and five real-world datasets) to compare the performance of our method with eight baselines. On each dataset, ten-fold cross-validation is performed and the mean accuracy, as well as standard deviation, are reported. In addition, we use a pairwise t-test at 0.05 significance level for two independent samples to investigate whether our method UCL is significantly superior/inferior (win/loss) to comparing methods.

Comparing Methods

To show the effectiveness of UCL, we compared it with eight existing PLL algorithms, each configured with suggested parameters in the respective literature.

- PL-KNN (Hüllermeier and Beringer 2006): a k-nearest neighbor approach that makes predictions by averaging the labeling information of neighboring examples [suggested configuration: $k \in \{5, 6, \cdots, 10\}$].
- PL-SVM (Nguyen and Caruana 2008): a maximum margin approach that learns from PL examples by optimizing margin-based objective function [suggested configuration: $\lambda \in \{10^{-3}, 10^{-2} \ldots, 10^{3}\}$].
- SURE (Feng and An 2019): a self-training based approach which trains the desired model and performs pseudo-labeling jointly by solving a convex-concave optimization problem [suggested configuration: regularization parameters $\lambda = 0.3, \beta = 0.05$].
- CENDA (Bao, Hang, and Zhang 2021): a dimensionality reduction approach via confidence-based dependence maximization which combines other algorithms such as SURE to work [suggested configuration: $thr = 0.999, \mu = 0.5, k = 8$].
- DELIN (Wu and Zhang 2019): a dimensionality reduction approach which adapts linear discriminant analysis and combines other algorithms such as SURE to work [suggested configuration: $r = 0.6, k = 8, \mathrm{T} = 75$].
- IPAL (Zhang and Yu 2015): an instance-based approach that disambiguates candidate labels by an adapted label propagation scheme [suggested configuration: $\alpha \in \{0, 0.1, \cdots, 1\}, k \in \{5, 6, \cdots, 10\}$].
- AGGD (Wang, Zhang, and Li 2021): a disambiguation approach which performs adaptive graph construction, candidate label disambiguation and predictive model induction with alternating optimization [suggested configuration: $k = 10, T = 20, \lambda = 1, \mu = 1, \gamma = 0.05$].
- PL-LE (Xu, Lv, and Geng 2019): an approach which considers generalized label distribution and learns from PL examples via label enhancement [suggested configuration: $k = 20, \lambda = 0.01, c1 = 1, c2 = 1$].

Results on Controlled UCI Datasets

Table 1 summarizes the characteristics of five UCI datasets (Dua and Graff 2017), ranging from small datasets to quite large datasets. Following the widely-used controlling protocol, an artificial partial label dataset is derived from one multi-class UCI dataset by configuring three controlling parameters p, r and ϵ (Cour, Sapp, and Taskar 2011; Liu and Dietterich 2012; Zhang and Yu 2015; Feng and An 2019; Wang, Zhang, and Li 2021). Here, p controls the proportion of examples that are partially labeled, r controls the number of false positive labels in the candidate label set, and ϵ is the probability of picking co-occurring label as a false positive label. As shown in Table 1, Configuration (I) exams varying r and Configuration (II) exams varying ϵ.

Table 2 illustrates the classification accuracy of each algorithm when $r = 1, 2,$ and 3 (Configuration (I)), respectively. In these three settings, r extra labels are randomly chosen to be the false positive labels. That is, the number of candidate labels for each instance is $r + 1$.

Figure 1 shows the classification accuracy of each algorithm as ϵ ranges from 0.1 to 0.7 when $p = 1$ and $r = 1$ (Configuration (II)). In this setting, a specific label is selected as the coupled label that co-occurs with the ground-truth label with probability ϵ, and any other label would be

Dataset	Examples	Features	Classes	Task Domain	Configurations
zoo	101	16	7	animal classification	
glass	214	9	6	glass classification	
yeast	1,484	8	10	yeast classification	(I) $p = 1, r \in \{1, 2, 3\}$
tmc2007	8,670	981	18	text anomaly detection	(II) $p = 1, r = 1, \epsilon \in \{0.1, 0.2, \cdots, 0.7\}$
sport	9,120	1738	19	human activity recognition	

Table 1: Characteristics of the controlled UCI datasets

Dataset	zoo	glass	yeast	tmc2007	sport
			$r = 1$ (one false positive label)		
UCL (*ours*)	0.9009 $\pm$ 0.1054	**0.7106** $\pm$ 0.0742	**0.5983** $\pm$ 0.0596	**0.6565** $\pm$ 0.0197	0.8978 $\pm$ 0.0116
PL-KNN	0.7936 $\pm$ 0.1318●	0.6409 $\pm$ 0.0941●	0.5755 $\pm$ 0.0494	0.4285 $\pm$ 0.0204●	0.7936 $\pm$ 0.1318●
PL-SVM	0.7736 $\pm$ 0.1374●	0.4820 $\pm$ 0.0942●	0.3450 $\pm$ 0.0544●	0.6449 $\pm$ 0.0152●	0.6820 $\pm$ 0.0151●
SURE	**0.9100** $\pm$ 0.1287	0.6784 $\pm$ 0.0873	0.5929 $\pm$ 0.0531	0.6451 $\pm$ 0.0170●	0.7601 $\pm$ 0.0157●
SURE-CENDA	0.8718 $\pm$ 0.1047	0.6314 $\pm$ 0.0971●	0.5977 $\pm$ 0.0486	0.6374 $\pm$ 0.0176●	0.7777 $\pm$ 0.0143●
SURE-DELIN	0.8909 $\pm$ 0.1102	0.6175 $\pm$ 0.1255●	0.5835 $\pm$ 0.0402	0.6074 $\pm$ 0.0164●	0.7456 $\pm$ 0.0092●
IPAL	0.8418 $\pm$ 0.1168●	0.6788 $\pm$ 0.1078	0.5027 $\pm$ 0.0603●	0.5945 $\pm$ 0.0210●	**0.9041** $\pm$ 0.0125
AGGD	0.8818 $\pm$ 0.1217	0.6833 $\pm$ 0.1067	0.5963 $\pm$ 0.0539	0.6459 $\pm$ 0.0162●	0.7776 $\pm$ 0.0135●
PL-LE	0.8909 $\pm$ 0.1198	0.6883 $\pm$ 0.1367	0.5970 $\pm$ 0.0496	0.6449 $\pm$ 0.0182	0.7498 $\pm$ 0.0164●
			$r = 2$ (two false positive labels)		
UCL (*ours*)	**0.8800** $\pm$ 0.1229	**0.6970** $\pm$ 0.1079	**0.5950** $\pm$ 0.0528	**0.6505** $\pm$ 0.0180	0.8860 $\pm$ 0.0044
PL-KNN	0.7736 $\pm$ 0.1105●	0.6455 $\pm$ 0.0996●	0.5519 $\pm$ 0.0488●	0.4057 $\pm$ 0.0228●	0.2993 $\pm$ 0.0181●
PL-SVM	0.6445 $\pm$ 0.1304●	0.4909 $\pm$ 0.0779●	0.3820 $\pm$ 0.0715●	0.6406 $\pm$ 0.0141●	0.6420 $\pm$ 0.0194●
SURE	0.8509 $\pm$ 0.1274	0.6645 $\pm$ 0.0904	0.5909 $\pm$ 0.0401	0.6383 $\pm$ 0.0162●	0.7200 $\pm$ 0.0149●
SURE-CENDA	0.8327 $\pm$ 0.1025	0.6126 $\pm$ 0.0897●	0.5936 $\pm$ 0.0353	0.6330 $\pm$ 0.0157●	0.7445 $\pm$ 0.0196●
SURE-DELIN	0.8327 $\pm$ 0.1025	0.6180 $\pm$ 0.1556●	0.5660 $\pm$ 0.0439●	0.6112 $\pm$ 0.0161●	0.7288 $\pm$ 0.0150●
IPAL	0.6155 $\pm$ 0.1728●	0.6872 $\pm$ 0.0597	0.4758 $\pm$ 0.0470●	0.5747 $\pm$ 0.0212●	**0.8977** $\pm$ 0.0123○
AGGD	0.8418 $\pm$ 0.1260	0.6597 $\pm$ 0.0856	0.5936 $\pm$ 0.0452	0.6374 $\pm$ 0.0140●	0.7636 $\pm$ 0.0139●
PL-LE	0.8518 $\pm$ 0.1428	0.6509 $\pm$ 0.1012	0.5876 $\pm$ 0.0459	0.6301 $\pm$ 0.0141●	0.7364 $\pm$ 0.0154●
			$r = 3$ (three false positive labels)		
UCL (*ours*)	**0.8318** $\pm$ 0.1248	**0.6649** $\pm$ 0.1227	0.5930 $\pm$ 0.0442	**0.6541** $\pm$ 0.0202	0.8711 $\pm$ 0.0102
PL-KNN	0.7545 $\pm$ 0.1350●	0.5574 $\pm$ 0.1074●	0.5088 $\pm$ 0.0504●	0.3789 $\pm$ 0.0216●	0.3013 $\pm$ 0.0183●
PL-SVM	0.4845 $\pm$ 0.1699●	0.1959 $\pm$ 0.0913●	0.3120 $\pm$ 0.0385●	0.6326 $\pm$ 0.0166●	0.6103 $\pm$ 0.0235●
SURE	0.8127 $\pm$ 0.0964	0.5749 $\pm$ 0.0917●	0.5916 $\pm$ 0.0390	0.6253 $\pm$ 0.0197●	0.6795 $\pm$ 0.0101●
SURE-CENDA	0.7736 $\pm$ 0.1105	0.5615 $\pm$ 0.0940●	0.5808 $\pm$ 0.0441●	0.6301 $\pm$ 0.0214●	0.7200 $\pm$ 0.0179●
SURE-DELIN	0.7436 $\pm$ 0.0913●	0.5524 $\pm$ 0.1055●	0.5680 $\pm$ 0.0450●	0.6069 $\pm$ 0.0176●	0.7088 $\pm$ 0.0129●
IPAL	0.3391 $\pm$ 0.1447●	0.5472 $\pm$ 0.1239●	0.4488 $\pm$ 0.0455●	0.5664 $\pm$ 0.0236●	**0.8922** $\pm$ 0.0137○
AGGD	0.7927 $\pm$ 0.0970	0.5898 $\pm$ 0.0990●	**0.5963** $\pm$ 0.0432	0.6290 $\pm$ 0.0181●	0.7523 $\pm$ 0.0154●
PL-LE	0.7727 $\pm$ 0.1144	0.5753 $\pm$ 0.0971●	0.5869 $\pm$ 0.0430	0.6232 $\pm$ 0.0192●	0.7206 $\pm$ 0.0141●

Table 2: Classification accuracy of each algorithm on the UCI datasets. Furthermore, ●/○ indicate UCL is statistically superior/inferior to the comparing algorithm (t-test at 0.05 significance level for two independent samples).

randomly chosen to be a false positive label with probability $1 - \epsilon$. As shown in Table 2 and Figure 1, UCL consistently outperforms other comparing algorithms. To further statistically compare UCL with other algorithms, the detailed win/tie/loss counts between UCL and the comparing algorithms are recorded in Table 3. From the above results, we have the following observations:

- According to Table 2, there is an indication that UCL achieves better performance against other algorithms when adding more false positive labels.
- UCL achieves superior or at least comparable performance against five algorithms including PL-KNN, PL-SVM, SURE-CENDA, SURE-DELIN and AGGD in all cases. Specifically, UCL outperforms them in 82%, 96%, 56%, 66% and 38% cases successively.
- UCL achieves superior performance against SURE, IPAL, and PL-LE in 42%, 60% and 42% cases, respectively.

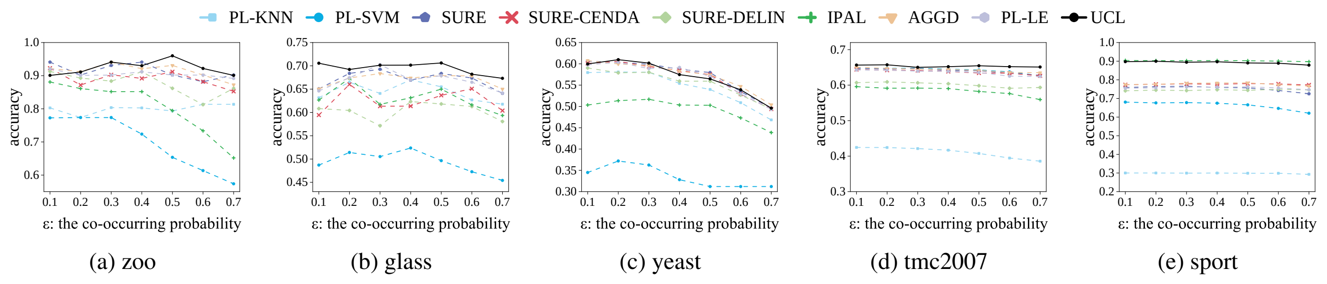

Figure 1: Classification performance on controlled UCI datasets with ϵ ranging from 0.1 to 0.7 ($p = 1, r = 1$).

Configuration	(I)	(II)	Total
PL-KNN	14/1/0	27/8/0	**41/9/0**
PL-SVM	15/0/0	33/2/0	**48/2/0**
SURE	7/8/0	14/20/1	**21/28/1**
SURE-CENDA	10/5/0	18/17/0	**28/22/0**
SURE-DELIN	12/3/0	21/14/0	**33/17/0**
IPAL	10/3/2	20/14/1	**30/17/3**
AGGD	7/8/0	12/23/0	**19/31/0**
PL-LE	6/9/0	15/19/1	**21/28/1**

Table 3: Win/tie/loss counts on UCI datasets of UCL.

Results on Real-World Datasets

In addition to five synthetic datasets, we further test our method on five real-world partial-labeled datasets which are collected from various task domains, i.e., Lost (Cour, Sapp, and Taskar 2011), MSRCv2 (Liu and Dietterich 2012), Mirflickr (Huiskes and Lew 2008), BirdSong (Briggs, Fern, and Raich 2012), and Soccer Player (Zeng et al. 2013). Table 4 shows the characteristics of real-world datasets.

Table 5 reports the mean classification accuracy and the standard deviation of each method on five real-world datasets. From the results, we can observe that:

- UCL outperforms PL-KNN, PL-SVM, SURE-DELIN and PL-LE significantly on all of the real-world datasets.
- Out of the 40 cases (8 comparing algorithms and 5 datasets), UCL is statistically superior to all the comparing algorithms in 85% cases and outperforms all the comparing algorithms in 95% cases.
- UCL is never significantly outperformed by any comparing algorithms in all experiments.

Further Understanding of Our Method

Parameter Sensitivity Analysis In the proposed UCL approach, three trade-off parameters τ_h, τ_l, and α need to be manually searched. Figure 2 shows how three parameters affect classification accuracy in four datasets (Lost, MSRCv2, glass, tmc2007), two of which are real-world datasets and another two are synthetic datasets. Specifically, we vary τ_l ranging from 0.1 to 0.4 when $\tau_h = 0.6, \alpha = 0.5$; vary τ_h ranging from 0.5 to 0.9 when $\tau_l = 0.2, \alpha = 0.5$; vary α ranging from 0.1 to 1.0 when $\tau_l = 0.2, \tau_h = 0.6$. As shown in Figure 2, the performance of UCL is generally stable

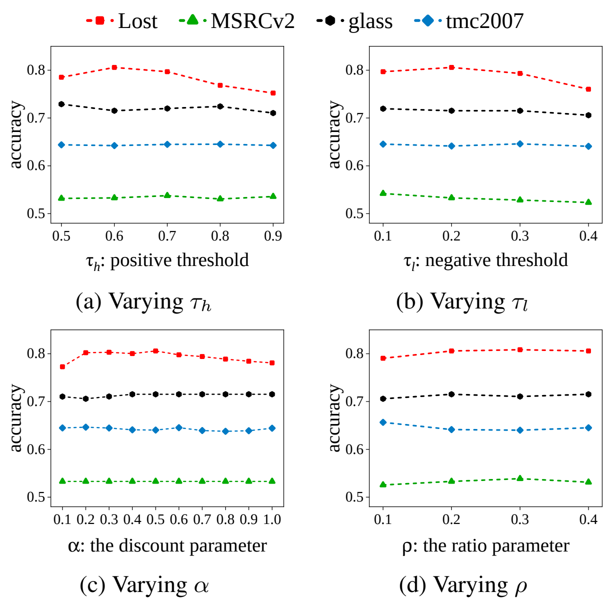

Figure 2: Parameter sensitivity studies.

across a wide range of each parameter, which means UCL achieves robust classification performance without costing much effort on parameter fine-tuning.

Ablation Studies on the Partitioning Strategy Notice that our method UCL splits the data for every class into three partitions, i.e., positive, negative, and uncertain. To verify the effectiveness and necessity of our partitioning strategy, we compare the classification accuracy of UCL with its two variants, i.e., UCL without a partition (denoted by "no partition") and UCL with only positive and negative partitions (denoted by "PN partition"). Specifically, in UCL without partition, we conduct spectral clustering on all training instances where cluster centers are utilized to produce label-specific features. Moreover, in PN PARTITION, we follow the original strategy of LIFT which partitions training instances into positive and negative ones for subsequent label-specific features generation. As mentioned above, LIFT solves multi-label learning problems where training examples can be classified into two parts according to their ground-truth labels while in PLL the partition is more complicated due to the inexact information. Thus, our

Dataset	Examples	Features	Classes	avg. CLs	Task Domain
Lost	1,122	108	16	2.23	automatic face naming (Cour, Sapp, and Taskar 2011)
MSRCv2	1,758	48	23	3.16	object classification (Liu and Dietterich 2012)
Mirflickr	2,780	1,536	14	2.76	web image classification (Huiskes and Lew 2008)
BirdSong	4,998	38	13	2.18	bird song classification (Briggs, Fern, and Raich 2012)
Soccer Player	17,472	279	171	2.09	automatic face naming (Zeng et al. 2013)

Table 4: Characteristics of the real-world partial label datasets.

Dataset	Lost	MSRCv2	Mirflickr	BirdSong	Soccer Player
UCL (*ours*)	**0.8084** $\pm$ 0.0378	**0.5387** $\pm$ 0.0265	<u>0.6716</u> $\pm$ 0.0182	<u>0.7377</u> $\pm$ 0.0254	**0.5531** $\pm$ 0.0112
PL-KNN	0.3877 $\pm$ 0.0317●	0.4539 $\pm$ 0.0339●	0.5011 $\pm$ 0.0270●	0.6473 $\pm$ 0.0237●	0.4943 $\pm$ 0.0103●
PL-SVM	0.7612 $\pm$ 0.0242●	0.3009 $\pm$ 0.0221●	0.5996 $\pm$ 0.0975●	0.5190 $\pm$ 0.0295●	0.4498 $\pm$ 0.0107●
SURE	<u>0.7808</u> $\pm$ 0.0292●	0.4733 $\pm$ 0.0220●	0.6691 $\pm$ 0.0210	**0.7403** $\pm$ 0.0379	0.5335 $\pm$ 0.0103●
SURE-CENDA	0.7781 $\pm$ 0.0487	0.4597 $\pm$ 0.0380●	0.3662 $\pm$ 0.0432●	0.6777 $\pm$ 0.0278●	0.5327 $\pm$ 0.0112●
SURE-DELIN	0.7727 $\pm$ 0.0316●	0.4392 $\pm$ 0.0341●	0.3716 $\pm$ 0.0339●	0.6403 $\pm$ 0.0365●	0.5316 $\pm$ 0.0093●
IPAL	0.7424 $\pm$ 0.0333●	<u>0.5301</u> $\pm$ 0.0300	0.5381 $\pm$ 0.0226●	0.7073 $\pm$ 0.0203●	<u>0.5499</u> $\pm$ 0.0106●
AGGD	0.7763 $\pm$ 0.0333●	0.5000 $\pm$ 0.0304●	**0.6745** $\pm$ 0.0254	0.7335 $\pm$ 0.0238	0.5434 $\pm$ 0.0101●
PL-LE	0.7576 $\pm$ 0.0373●	0.5068 $\pm$ 0.0243●	0.6414 $\pm$ 0.0249●	0.7235 $\pm$ 0.0285●	0.5360 $\pm$ 0.0200●

Table 5: Classification accuracy of each algorithm on the real-world datasets.

Dataset	Lost	MSRCv2	Mirflickr	BirdSong	Soccer Player
UCL-KNN (*ours*)	**0.5312** $\pm$ 0.0665	0.4221 $\pm$ 0.0452	**0.5043** $\pm$ 0.0354	0.6413 $\pm$ 0.0250	**0.4998** $\pm$ 0.0106
PL-KNN	0.3877 $\pm$ 0.0317●	**0.4539** $\pm$ 0.0339○	0.5011 $\pm$ 0.0270	**0.6473** $\pm$ 0.0237	0.4943 $\pm$ 0.0103●
UCL-SVM (*ours*)	**0.7656** $\pm$ 0.0431	**0.3937** $\pm$ 0.0405	0.5453 $\pm$ 0.0261	**0.5938** $\pm$ 0.0268	**0.4742** $\pm$ 0.0134
PL-SVM	0.7612 $\pm$ 0.0242	0.3009 $\pm$ 0.0221●	**0.5996** $\pm$ 0.0975	0.5190 $\pm$ 0.0295●	0.4498 $\pm$ 0.0107●

Table 6: Classification accuracy of combining UCL with KNN or SVM on real-world datasets.

experiments on PN PARTITION employ the confidence partitioning strategy of UCL while only reserving the parameter $\tau_h = 0.6$ to divide training data into two parts. We conduct experiments on three synthetic datasets where one false positive label is randomly picked up (i.e., $r = 1$) as well as three real-world datasets. As shown in Figure 3, our partitioning strategy makes successful progress on each dataset.

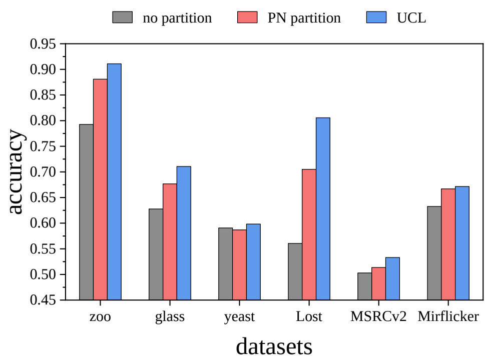

Figure 3: Ablation studies on the partitioning strategy.

Extension to Other Classifiers In Table 6, we test the combination of UCL with another two popular PLL classifiers, i.e., UCL-KNN and UCL-SVM, on five real-world datasets. We can observe that in the t-test at 0.05 significance level UCL-KNN achieves $2/2/1$ (win/tie/loss) compared to PL-KNN, and UCL-SVM achieves $3/2/0$ (win/tie/loss) compared to PL-SVM. This indicates that there is still much room for improving the versatility of our approach. Notice that our default choice of classifier, i.e., SURE, is more powerful than *k*-NN and SVM, we recommend combining label-specific features with strong classifiers.

Conclusion

This paper for the first time studies the utility of label-specific features for PLL. It answers the fundamental question “Can label-specific features help PLL?” in the affirmative. Our proposed approach UCL consistently improves the performance on both synthetic datasets and real-world datasets. However, when combined with weak classifiers, the positive effect of label-specific features is unnoticeable, and the performance may even deteriorate. We believe this paper can motivate more research by generating more informative label-specific features and combining them with more powerful classifiers such as deep neural networks.

References

Bao, W.-X.; Hang, J.-Y.; and Zhang, M.-L. 2021. Partial label dimensionality reduction via confidence-based dependence maximization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 46–54.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9): 509–517.

Briggs, F.; Fern, X. Z.; and Raich, R. 2012. Rank-loss support instance machines for MIML instance annotation. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 534–542.

Chai, J.; Tsang, I. W.; and Chen, W. 2019. Large margin partial label machine. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7): 2594–2608.

Chen, C.-H.; Patel, V. M.; and Chellappa, R. 2017. Learning from ambiguously labeled face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(7): 1653–1667.

Chen, Y.-C.; Patel, V. M.; Chellappa, R.; and Phillips, P. J. 2014. Ambiguously labeled learning using dictionaries. *IEEE Transactions on Information Forensics and Security*, 9(12): 2076–2088.

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

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Feng, L.; and An, B. 2019. Partial label learning with self-guided retraining. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 3542–3549.

Feng, L.; Lv, J.; Han, B.; Xu, M.; Niu, G.; Geng, X.; An, B.; and Sugiyama, M. 2020. Provably consistent partial-label learning. *Advances in Neural Information Processing Systems 33*, 10948–10960.

Gong, C.; Liu, T.; Tang, Y.; Yang, J.; Yang, J.; and Tao, D. 2017. A regularization approach for instance-based superset label learning. *IEEE Transactions on Cybernetics*, 48(3): 967–978.

Hang, J.; Zhang, M.; Feng, Y.; and Song, X. 2022. End-to-end probabilistic label-specific feature learning for multi-label Classification. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 6847–6855.

Huang, J.; Li, G.; Huang, Q.; and Wu, X. 2015. Learning label specific features for multi-label classification. In *Proceedings of the 15th International Conference on Data Mining*, 181–190.

Huang, J.; Li, G.; Huang, Q.; and Wu, X. 2017. Joint feature selection and classification for multilabel learning. *IEEE Transactions on Cybernetics*, 48(3): 876–889.

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.

Hüllermeier, E.; and Beringer, J. 2006. Learning from ambiguously labeled examples. *Intelligent Data Analysis*, 10(5): 419–439.

Jin, R.; and Ghahramani, Z. 2002. Learning with multiple labels. *Advances in Neural Information Processing Systems 15*, 897–904.

Kashef, S.; and Nezamabadi-pour, H. 2019. A label-specific multi-label feature selection algorithm based on the Pareto dominance concept. *Pattern Recognition*, 88: 654–667.

Liu, L.; and Dietterich, T. 2012. A conditional multinomial mixture model for superset label learning. *Advances in Neural Information Processing Systems 25*, 557–565.

Luo, J.; and Orabona, F. 2010. Learning from candidate labeling sets. *Advances in Neural Information Processing Systems 23*, 1504–1512.

Lv, J.; Xu, M.; Feng, L.; Niu, G.; Geng, X.; and Sugiyama, M. 2020. Progressive identification of true labels for partial-label learning. In *Proceedings of the 37th International Conference on Machine Learning*, 6500–6510.

Lyu, G.; Feng, S.; Wang, T.; and Lang, C. 2020. A self-paced regularization framework for partial-label learning. *IEEE Transactions on Cybernetics*, 52(2): 899–911.

Nguyen, N.; and Caruana, R. 2008. Classification with partial labels. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 551–559.

Shi, J.; and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8): 888–905.

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.

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

Wang, D.-B.; Zhang, M.-L.; and Li, L. 2021. Adaptive graph guided disambiguation for partial label learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, in press.

Wang, H.; Xiao, R.; Li, Y.; Feng, L.; Niu, G.; Chen, G.; and Zhao, J. 2022. PiCO: Contrastive label disambiguation for partial label learning. *International Conference on Learning Representations*.

Wei, T.; Tu, W.; and Li, Y. 2019. Learning for tail label data: A label-specific feature approach. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 3842–3848.

Wen, H.; Cui, J.; Hang, H.; Liu, J.; Wang, Y.; and Lin, Z. 2021. Leveraged weighted loss for partial label learning. In *Proceedings of the 38th International Conference on Machine Learning*, 11091–11100.

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, J.-H.; and Zhang, M.-L. 2019. Disambiguation enabled linear discriminant analysis for partial label dimensionality reduction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 416–424.

Xu, N.; Lv, J.; and Geng, X. 2019. Partial label learning via label enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5557–5564.

Xu, S.; Yang, X.; Yu, H.; Yu, D.-J.; Yang, J.; and Tsang, E. C. 2016. Multi-label learning with label-specific feature reduction. *Knowledge-Based Systems*, 104: 52–61.

Yu, F.; and Zhang, M.-L. 2016. Maximum margin partial label learning. In *Proceedings of the 8th Asian Conference on Machine Learning*, 96–111.

Yu, Z.-B.; and Zhang, M.-L. 2021. Multi-label classification with label-specific feature generation: A wrapped approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, in press.

Zeng, Z.; Xiao, S.; Jia, K.; Chan, T.-H.; Gao, S.; Xu, D.; and Ma, Y. 2013. Learning by Associating Ambiguously Labeled Images. In *Proceedings of the 26th Conference on Computer Vision and Pattern Recognition*, 708–715.

Zhang, J.; Li, C.; Cao, D.; Lin, Y.; Su, S.; Dai, L.; and Li, S. 2018. Multi-label learning with label-specific features by resolving label correlations. *Knowledge-Based Systems*, 159: 148–157.

Zhang, J.-J.; Fang, M.; and Li, X. 2015. Multi-label learning with discriminative features for each label. *Neurocomputing*, 154: 305–316.

Zhang, M.-L.; and Wu, L. 2014. Lift: Multi-label learning with label-specific features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1): 107–120.

Zhang, M.-L.; and Yu, F. 2015. Solving the partial label learning problem: An instance-based approach. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, 4048–4054.

Zhang, M.-L.; Zhou, B.-B.; and Liu, X.-Y. 2016. Partial label learning via feature-aware disambiguation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1335–1344.

Zhang, M.-L.; and Zhou, Z.-H. 2013. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8): 1819–1837.

Zhou, D.; Zhang, Z.; Zhang, M.-L.; and He, Y. 2018. Weakly supervised POS tagging without disambiguation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 17(4): 1–19.

Zhou, Z.-H. 2018. A brief introduction to weakly supervised learning. *National Science Review*, 5(1): 44–53.