Adaptive Graph Guided Disambiguation for Partial Label Learning

Deng-Bao Wang, Min-Ling Zhang, and Li Li

Abstract—In partial label learning, a multi-class classifier is learned from the ambiguous supervision where each training example is associated with a set of candidate labels among which only one is valid. An intuitive way to deal with this problem is *label disambiguation*, i.e. differentiating the labeling confidences of different candidate labels so as to try to recover ground-truth labeling information. Recently, feature-aware label disambiguation has been proposed which utilizes the graph structure of feature space to generate labeling confidences over candidate labels. Nevertheless, the existence of noises and outliers in training data makes the graph structure derived from original feature space less reliable. In this paper, a novel partial label learning approach based on *adaptive graph guided disambiguation* is proposed, which is shown to be more effective in revealing the intrinsic manifold structure among training examples. Other than the sequential *disambiguation-then-induction* learning strategy, the proposed approach jointly performs adaptive graph construction, candidate label disambiguation and predictive model induction with alternating optimization. Furthermore, we consider the particular *human-in-the-loop* framework in which a learner is allowed to actively query some ambiguously labeled examples for manual disambiguation. Extensive experiments clearly validate the effectiveness of adaptive graph guided disambiguation for learning from partial label examples.

Index Terms—Machine Learning, weakly supervised learning, partial label learning, label disambiguation.

1 Introduction

Multi-class classification has been successfully applied in various real-world applications such as computer vision, natural language processing and web advertising [11], [71]. It is well-known that learning from supervised data sets is effective, however, collecting large-scale accurately labeled data is expensive and thus a critical bottleneck in many tasks. To solve this problem, weakly supervised learning has been widely studied in recent years [9], [21], [26], [29], [30], [40], [41], [53], [73], [75]. Partial label learning is an emerging weakly-supervised learning framework for learning from ambiguous supervision [6], [7], [14], [15], [18], [38], [44], [45], [48]–[51], [54]–[57], [64], [68], [69]. It aims to induce a multi-class classifier from training examples where each of them is associated with a set of candidate labels, among which only one is the ground-truth label. This learning problem arises in domains in which a large number of ambiguously labeled (or weakly supervised) training examples are available while it is costly to acquire explicit labeled data [10], [32], [35], [65].

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ be the d dimensional feature space and $\mathcal{Y} = \{0, 1\}^q$ be the label space with q possible class labels. Suppose we have the partial label training set $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i) | \ 1 \leq i \leq m\}$ where $\boldsymbol{x}_i \in \mathcal{X}$ is a d dimensional feature vector and Y_i is the corresponding candidate label set. The task of partial label learning is to induce a multi-class classifier $g : \mathcal{X} \to \mathcal{Y}$ based on training set $\mathcal{D}$. Here the basic assumption under partial label learning framework is that the ground truth label y_i of $\boldsymbol{x}_i$ resides in Y_i but it is not accessible to the algorithm during training phase.

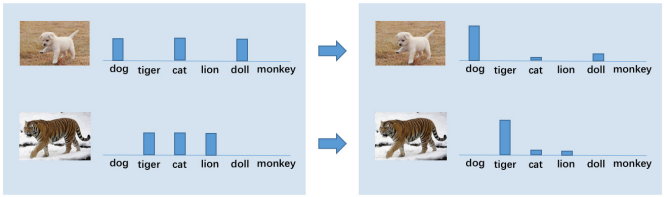


Fig. 1. Label disambiguation for partially labeled examples. Left: The instances are assigned with multiple candidate labels. Right: The ground-truth labeling confidences are expected to be recovered after label disambiguation.

The major difficulty for partial label learning lies in that the ground-truth label of each training example is not directly accessible to the learning algorithm while concealed in its candidate label set. Therefore, the common strategy to learn from partial labeled data is to try to disambiguate the candidate label sets, i.e. differentiating the labeling confidences of individual candidate labels so as to recover ground-truth labeling information (Fig. 1). There are mainly two label disambiguation strategies, i.e. the averaging-based strategy and the identification-based strategy. For averaging-based disambiguation, all the potential labels of each example are treated equally and the averaged output from all candidate labels is distinguished from the outputs from non-candidate labels [9], [25], [67]. For identification-based disambiguation, the ground-truth label is regarded as latent variable and identified through iterative refining procedure such as EM algorithm [8], [27], [32], [36], [38].

Feature-aware disambiguation [64], [69] was proposed

- *Deng-Bao Wang, and Min-Ling Zhang are with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. E-mail: {wangdb, zhangml}@seu.edu.cn.*
- *Li Li is with the College of Computer and Information Science, Southwest University, Chongqing 400715, China, E-mail: lily@swu.edu.cn.*

(a) POS tagging. (b) Automatic face naming. (c) Learning with partial annotation.

Fig. 2. There real-world partial label learning scenarios. (a) In Part-of-speech (POS) tagging, the predictive model will be learned given a dictionary of words with their possible POS tags; (b) In automatic face naming, the predictive model will be learned with the image/caption pair collected from websites; (c) In crowdsourcing-based classification, the predictive model will be learned from inaccurate labeling information.

recently to generate labeling confidences over candidate label set by exploiting the manifold structure of instance space. Feature-aware disambiguation makes use of the smoothness assumption to facilitate the disambiguation process, i.e. the examples close to each other in the feature space will tend to share identical label in the label space. Existing feature-aware label disambiguation approaches work in a two-stage manner which firstly generate pseudo labeling confidences over candidate label set for each example and then learn a multi-class classifier by fitting a multi-output regressor with the generated labeling confidences.

The results reported in previous partial label learning studies [13], [67], [69] have shown the effectiveness of feature-aware disambiguation strategy. Nevertheless, the existence of noises and outliers in training data makes the graph structure derived from original feature space less reliable. To solve this potential drawback, we propose a novel partial label learning approach in this paper based on *adaptive graph guided disambiguation* (PL-AGGD). We expect that adaptive graph guided disambiguation could be more robust and accurate to reveal the intrinsic manifold structure among training examples compared to fixed graph based disambiguation. Instead of the two-stage learning strategy used in previous approaches, our proposed approach aims to accomplish the partial label learning task in a joint manner. Specifically, a unified optimization framework was presented which enables simultaneous adaptive graph construction, candidate label disambiguation and predictive model induction with alternating optimization. Extensive experiments on controlled UCI data sets as well as real-world partial label data sets show that PL-AGGD performs favorably against other state-of-the-art partial label learning approaches.

Furthermore, we consider the particular *human-in-the-loop* framework in which a learner is allowed to selectively query examples from ambiguously labeled data for manual disambiguation. Two simple yet effective query strategies are proposed which can be conveniently injected into our PL-AGGD method and significantly improve the predictive performance with small amount of queries and annotations. We also empirically show that the superiority of the proposed method compared with the fixed-graph-based counterpart is more obvious when active disambiguation is allowed in the loop.

The rest of the paper is organized as follows. We review related work in Section 2, and introduce the proposed approach in Section 3; Section 4 presents the settings and results of the experiments, followed by the conclusion in Section 5.

2 Related Work

Real-World Applications In recent years, the need to learn from ambiguous supervision naturally arises in many real-world applications. For instance, a Part-of-speech (POS) tagging model can be learned given a dictionary of words with their possible POS tags among which only one tag is true (Fig. 2a) [72]; in automatic face naming, for a news article one can treat each face detected from the picture as an instance and those names extracted from associated caption as candidate labels, while the ground-truth correspondence between each face and name is not known (Fig. 2b) [20], [65]; for a crowdsourcing-based classification, if the label space is large (can be hundreds or even thousands of labels), it is hard for the annotator to select the exact ground-truth label among all possible labels while relatively easy to query inaccurate labeling information (Fig. 2c).

Relations to Other Frameworks In partial label learning, the ground-truth label of a training example resides in its candidate label set and thus is not accessible to the algorithm during training phase. Therefore, partial label learning can be considered as a special weakly-supervised learning framework with ambiguous supervision. To put the partial label learning problem into perspective, we firstly lay out several related machine learning scenarios, including semi-supervised learning, multi-instance learning, multi-label learning and multi-instance multi-label learning. In semi-supervised learning [75], training examples are either explicitly labeled or unlabeled, while in partial label learning training examples are ambiguously labeled. Multi-instance learning [1], [2] learns from examples which are not individually labeled but grouped into bags, while in partial label learning labels are assigned at the level of individual instances. Multi-label learning [17], [33], [34], [46], [70] learns from data sets in which each example is assigned with multiple labels all of which are valid, while in partial label learning the set of labels assigned to training examples are only candidate ones. Multi-instance multi-label learning

[37], [74] can be regarded as a generalized version of multi-instance learning and multi-label learning, where training instances are not only represented by multiple instances but also associated with multiple labels. Each multi-instance multi-label example can be transformed into a number of partial label examples by treating the assigned class labels as candidate ones for each instance in the bag [32].

Partial Label Disambiguation In recent years, many partial label learning approaches have been proposed. The major strategy is trying to solve the learning task by disambiguating the candidate label set, which can be achieved in two basic ways. One intuitive way is to treat all the candidate labels of an example in an equal manner. For parametric model $g(\boldsymbol{x})$ $(y \in \mathcal{Y})$, the averaged output over all candidate labels, i.e. $\frac{1}{|Y_i|}\sum_{y\in Y_i} g_y(\boldsymbol{x}_i)$, is distinguished from the outputs from non-candidate labels, i.e. $g_y(\boldsymbol{x}_i)$ $(y \notin Y_i)$ [9]; For instance-based models, the predicted label for test example is determined by voting among the candidate labels of its neighboring examples [25]. Despite the intuitiveness of the averaging-based strategy, its effectiveness could be affected by the false positive labels whose outputs would overwhelm the essential output yielded by the ground-truth label. Therefore, another way towards disambiguating the candidate label set is to identify the ground-truth label along with model training. Existing approaches following this strategy take the ground-truth label as latent variable which is identified as: $\hat{y}_i = \arg\max_{y\in Y_i} g_y(\boldsymbol{x}_i)$. Generally, the latent variable and model parameters are refined iteratively via EM-style procedure which optimizes objective function defined according to the maximum likelihood criterion: $\sum_{i=1}^{m}\log(\sum_{y\in Y_i} g_y(\boldsymbol{x}_i))$ [27], [32], or the maximum margin criterion: $\sum_{i=1}^{m}(\max_{y\in Y_i} g_y(\boldsymbol{x}_i) - \max_{y\notin Y_i} g_y(\boldsymbol{x}_i))$ [38], [63]. The drawback of the identification-based disambiguation strategy lies in that, rather than recovering the ground-truth label, the identified label might turn out to be false positive label in the candidate label set.

Graph-Based Disambiguation The recently-proposed feature-aware disambiguation strategy [13], [67], [69] makes use of the local manifold structure in feature space to help disambiguate the candidate label set, which differs from both averaging-based and identification-based disambiguation strategies. In [67], an instance-based method was proposed which firstly disambiguates the label space using the manifold structure of instance space, and then predicts the unknown examples by label propagation. The approach proposed in [69] works in a two-stage manner which firstly generates latent labeling confidences over candidate label set by utilizing the graph structure of feature space and then learns a multi-class model by fitting a multi-output regressor with the generated labeling confidences. Another approach proposed in [13] also works in a two-stage manner by firstly constructing the graph structure and then leveraging the latent label distributions for model training. One potential drawback of this strategy lies in that real-world data are usually contaminated by significant noises and outliers which make the fixed graph structure recovered from original feature space less reliable [52], [66]. In the next section, we will introduce a novel approach which can iteratively refine the labeling confidences and model parameters while update the similarity graph adaptively.

TABLE 1
Summary of major mathematical notations.

Notations	Mathematical Meanings
$\mathcal{X}, \mathcal{Y}$	feature and label spaces
$D, \widetilde{D}$	ambiguous data set and disambiguated data set
d, q	dimensions of feature and label spaces
$\boldsymbol{x}_i, y_i$	feature and label of the i-th example
Y_i	candidate label set of the i-th example
$\mathbf{X}, \mathbf{Y}$	feature matrix and partial label matrix
$\mathcal{G}, \mathcal{V}, \mathcal{E}$	similarity graph and its vertices and edges
$\mathbf{S}$	weight matrix of similarity graph
$\mathbf{F}$	disambiguated label confidence matrix
$\mathbf{W}$	model parameter matrix
$\mathbf{H}$	model output matrix
$\mathbf{K}$, $\kappa(\bullet,\bullet)$	kernel matrix and kernel function
$\phi(\bullet)$	mapping function to Hilbert space
$\mathbf{G}^f, \mathbf{G}^x, \mathbf{G}^x_\phi, \mathbf{\Lambda}$	different gram matrices

Active Learning with Weak Supervision In our work, we are also interested in the active learning with partially labeled data sets. There are many papers studying the combination of active learning and weakly supervised learning [3], [4], [12], [19], [22], [23], [31], [43], [58]–[61]. Most of these literatures focus on utilizing active learning techniques when learning from imperfect supervision like crowdsourced data sets. The most similar setting with our work is learning with partial feedback [22], in which a learner actively chooses instance-label pairs and obtain the correctness of their correlations (as is shown in Fig. 2c), then iteratively learns from these partial feedback. In this paper, we show that our adaptive graph guided label disambiguation method can be conveniently injected into this human-in-loop framework, and the active disambiguation makes our method more effective with the help of manual disambiguation.

3 The Proposed Approach

In this subsection, we firstly present a unified framework which aims to perform candidate label disambiguation, adaptive graph construction and predictive model induction simultaneously. Then, we introduce an alternative optimization algorithm to solve the proposed objective function. We also extent our method with kernelization for general nonlinear cases. Finally, we discuss the active learning scenario of partial label learning tasks and show that our method can be easily extended to this setting.

3.1 Symbol Definitions and Notations

Denote $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m]^\top \in \mathbb{R}^{m\times d}$ as the feature matrix and $\mathbf{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_m]^\top \in \{0,1\}^{m\times q}$ as the partial label matrix where $y_{ij} = 1$ means that the j-th label is one of the candidate labels of $\boldsymbol{x}_i$. Given the training set $\mathcal{D}$, a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$ is constructed over the training examples, where $\mathcal{V} = \{\boldsymbol{x}_i | 0 \le i \le m\}$ corresponds to the set of vertices and $\mathcal{E} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) | \boldsymbol{x}_i \in KNN(\boldsymbol{x}_j), i \neq j\}$ corresponds to the set of edges from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ iff $\boldsymbol{x}_i$ is among the k-nearest neighbors of $\boldsymbol{x}_j$. Furthermore, $\mathbf{S} \in \mathbb{R}^{m\times m}$ corresponds to the non-negative weight matrix where $s_{ij} = 1$

if $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}$ and $s_{ij} = 1$ otherwise. For ease of reference, Table 1 lists major notations used throughout this section along with their mathematical meanings.

3.2 Disambiguation with Fixed Graph

Given an ambiguously labeled training data set $\mathcal{D}$, we aim to generate a normalized real-valued labeling confidence matrix $\mathbf{F} \in \mathbb{R}^{m \times q}$. For each training example $(\boldsymbol{x}_i, Y_i)$, the corresponding normalized real-valued vector is denoted as $\boldsymbol{f}_i \in \mathbb{R}^q$, where each f_{il} represents the labeling confidence of the l-th label being the ground-truth label for $\boldsymbol{x}_i$. The labeling confidence vector $\boldsymbol{f}_i$ satisfies the following constraints: (i) $\sum_{y_{il}=1} f_{il} = 1$ (normalization), (ii) $f_{il} \geq 0$ $(\forall y_{il} = 1)$ (non-negativity), and (iii) $f_{il} = 0$ $(\forall y_{il} = 0)$. The second constraint implies that the ground-truth label of each example resides in the candidate label set, and the third constraint guarantees that the labeling confidence of each non-candidate label must be 0. Note that $\mathbf{F}$ is a learnable matrix and we can learn this matrix by minimizing the following Eq. (2) once the similarity graph weights $\mathbf{S}$ is determined.

After generating the labeling confidence matrix $\mathbf{F}$, the original partial label data set $\mathcal{D}$ can be transformed into its disambiguated counterpart: $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, \boldsymbol{f}_i) | 1 \leq i \leq m\}$. Then we can induce the predictive model by performing a multi-output regression on the disambiguated data set $\widetilde{\mathcal{D}}$. In feature-aware disambiguation, the disambiguated labeling confidences are generated by exploiting the graph structure of the feature space. Specifically, the similarity graph weight matrix $\mathbf{S}$ can be constructed by solving the following linear least square problem:

$$\begin{aligned} \min_{\mathbf{S}} \ & \sum_{j=1}^{m} \left\| \boldsymbol{x}_j - \sum\nolimits_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}} s_{ij} \boldsymbol{x}_i \right\|_2^2 \\ \text{s.t.} \ & \mathbf{S}^\top \mathbf{1}_m = \mathbf{1}_m, \\ & s_{ij} \geq 0 \ (\forall (\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}), \ s_{ij} = 0 \ (\forall (\boldsymbol{x}_i, \boldsymbol{x}_j) \notin \mathcal{E}), \end{aligned} \tag{1}$$

where $\mathbf{1}_m$ is an all 1 vector with size m. After obtaining the graph weight matrix, the labeling confidence matrix $\mathbf{F}$ can be acquired by solving the following problem:

$$\begin{aligned} \min_{\mathbf{F}} \ & \sum_{j=1}^{m} \left\| \boldsymbol{f}_j - \sum\nolimits_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}} s_{ij} \boldsymbol{f}_i \right\|_2^2 \\ \text{s.t.} \ & \mathbf{F}\mathbf{1}_q = \mathbf{1}_m, \ f_{il} \geq 0 \ (\forall y_{il} = 1), \ f_{il} = 0 \ (\forall y_{il} = 0). \end{aligned} \tag{2}$$

This optimization problem is formalized according to the smoothness assumption that the similarity of instance in the feature space should also be preserved in the label space.

3.3 Adaptive Graph Guided Disambiguation

The above two optimization problems are solved independently in previous feature-aware label disambiguation approaches, thus the disambiguation process is actually guided by a fixed graph. It is challenging to reveal the intrinsic structure within the data using the fixed similarity graph due to the widely-existing of noises and outliers. A natural idea to solve this issue is to consider generating the similarity weight matrix in an adaptive way. Therefore, we choose to simultaneously generate the similarity weight matrix and differentiate the labeling confidence so as to achieve better disambiguation results. By solving the above two problems jointly, the similarity graph weights can be not only determined by the similarity information of features, but also adjusted based on the feedback from label space. Compared to fixed graph, adaptive graph could be more robust and accurate to reveal the similarity and intrinsic manifold structure within the data.

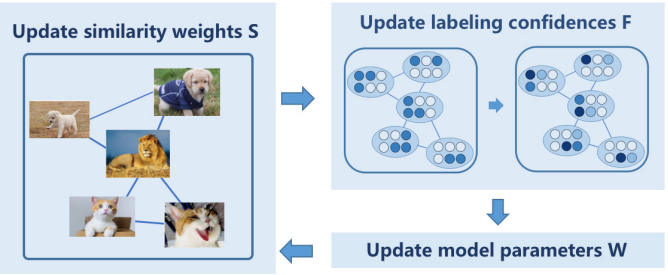


Fig. 3. The optimization procedure of our PL-AGGD approach. From this illustration we can see in our approach, the similarity weights between examples are dynamically updated with the model training and label disambiguation.

Instead of using the two-stage (disambiguation-then-induction) strategy in previous algorithms [64], [69], we further perform candidate label disambiguation and predictive model training simultaneously. Thus the labeling confidence can be optimized by leveraging the manifold structure in feature space and modeling outputs in label space. To this end, we present a unified optimization framework to make labeling confidences, similarity weights and predictive model parameters be jointly optimized to achieve strong generalization performance. We denote matrix $\mathbf{W} \in \mathbb{R}^{d \times q}$ as the model parameters, and use the least squares loss to optimize the predictive model:

$$\ell(g(\boldsymbol{x}_i), \boldsymbol{f}_i) = \left\| \mathbf{W}^\top \boldsymbol{x}_i - \boldsymbol{f}_i \right\|_2^2. \tag{3}$$

And we adopt the widely-used squared Frobenius norm as the regularization term to control the model complexity. Then the objective function is shown as below:

$$\begin{aligned} \min_{\mathbf{S},\mathbf{F},\mathbf{W}} \ & \sum_{j=1}^{m} \left\| \mathbf{W}^\top \boldsymbol{x}_i - \boldsymbol{f}_i \right\|_2^2 + \mu \sum_{j=1}^{m} \left\| \boldsymbol{f}_j - \sum\nolimits_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}} s_{ij} \boldsymbol{f}_i \right\|_2^2 \\ & + \gamma \sum_{j=1}^{m} \left\| \boldsymbol{x}_j - \sum\nolimits_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}} s_{ij} \boldsymbol{x}_i \right\|_2^2 + \lambda \left\| \mathbf{W} \right\|_{\mathrm{F}}^2 \\ \text{s.t.} \ & \mathbf{S}^\top \mathbf{1}_m = \mathbf{1}_m, \ \mathbf{0}_{m \times m} \leq \mathbf{S} \leq \mathbf{N}, \\ & \mathbf{F}\mathbf{1}_q = \mathbf{1}_m, \ \mathbf{0}_{m \times q} \leq \mathbf{F} \leq \mathbf{Y}, \end{aligned} \tag{4}$$

where $\mathbf{N} \in \{0, 1\}^{m \times m}$ is defined as: $n_{ij} = 1$ if $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}$ and $n_{ij} = 0$ otherwise. Furthermore, $\mathbf{0}_{m \times m}$ is the $m \times m$ all 0 matrix, λ is the trade-off parameter of the regularization term, and μ, γ are the trade-off parameters for reconstruction loss in label space and feature space respectively.

3.4 Alternative Optimization

As shown in previous subsection, the joint optimization problem (4) contains three sets of variables with different regularizations and constraints, thus it is hard to be tackled directly. In this subsection, we show that this problem can be solved by applying the alternative optimization algorithm.

Specifically, each set of variables will be iteratively optimized by fixing other sets of variables until convergence or the maximum number of iterations is reached.

Update S With fixed $\mathbf{F}$ and $\mathbf{W}$, we can re-write the optimization problem (4) as follows:

$$\begin{aligned}\min_{\mathbf{S}} \ & \mu\sum_{j=1}^{m}\left\|\boldsymbol{f}_j-\sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{f}_i\right\|_2^2 \\ & +\gamma\sum_{j=1}^{m}\left\|\boldsymbol{x}_j-\sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{x}_i\right\|_2^2 \\ \text{s.t. } & \mathbf{S}^\top\mathbf{1}_m=\mathbf{1}_m,\ \mathbf{0}_{m\times m}\le\mathbf{S}\le\mathbf{N}. \end{aligned} \tag{5}$$

Since the similarity vector of each sample is independent with others (i.e. each column of similarity matrix $\mathbf{S}$ is independent with other columns), we can fix other variables and optimize each similarity vector one by one. We consider to solve the following problem for the j-th sample in the rest of this subsection:

$$\begin{aligned}\min_{\mathbf{S}_{\cdot j}} \ & \mu\left\|\boldsymbol{f}_j-\sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{f}_i\right\|_2^2+\gamma\left\|\boldsymbol{x}_j-\sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{x}_i\right\|_2^2 \\ \text{s.t. } & \mathbf{S}_{\cdot j}^\top\mathbf{1}_m=1,\ \mathbf{0}_m\le\mathbf{S}_{\cdot j}\le\mathbf{N}_{\cdot j}, \end{aligned} \tag{6}$$

where $\mathbf{S}_{\cdot j}$ is the j-th column of $\mathbf{S}$, and it corresponds to the reconstruction weights for $\boldsymbol{x}_j$. Note that for $\mathbf{S}_{\cdot j}$, there are only k non-negative elements which we need to update. Let $\hat{\boldsymbol{s}}_j\in\mathbb{R}^k$ denotes the similarity weight vector with each element representing a weight value that characterizes the relative importance of a neighboring sample in reconstructing $\boldsymbol{x}_j$. Denote matrix $\mathbf{D}^{f_j}=[\boldsymbol{f}_j-\boldsymbol{f}_{\mathcal{N}_j(1)},\boldsymbol{f}_j-\boldsymbol{f}_{\mathcal{N}_j(2)},...,\boldsymbol{f}_j-\boldsymbol{f}_{\mathcal{N}_j(k)}]^\top\in\mathbb{R}^{k\times q}$ and matrix $\mathbf{D}^{x_j}=[\boldsymbol{x}_j-\boldsymbol{x}_{\mathcal{N}_j(1)},\boldsymbol{x}_j-\boldsymbol{x}_{\mathcal{N}_j(2)},...,\boldsymbol{x}_j-\boldsymbol{x}_{\mathcal{N}_j(k)}]^\top\in\mathbb{R}^{k\times d}$. Then the optimization problem (6) can be re-written as follows:

$$\begin{aligned}\min_{\hat{\boldsymbol{s}}_j} \ & \hat{\boldsymbol{s}}_j^\top(\mu\mathbf{G}^{f_j}+\gamma\mathbf{G}^{x_j})\hat{\boldsymbol{s}}_j \\ \text{s.t. } & \hat{\boldsymbol{s}}_j^\top\mathbf{1}_k=1,\ \mathbf{0}_k\le\hat{\boldsymbol{s}}_j\le\mathbf{1}_k, \end{aligned} \tag{7}$$

where $\mathbf{G}^{f_j}=\mathbf{D}^{f_j}(\mathbf{D}^{f_j})^\top\in\mathbb{R}^{k\times k}$ and $\mathbf{G}^{x_j}=\mathbf{D}^{x_j}(\mathbf{D}^{x_j})^\top\in\mathbb{R}^{k\times k}$ are two Gram matrices corresponding to label space and feature space respectively. The optimization problem (7) is a standard Quadratic Programming (QP) problem with only k variables which can be efficiently solved by off-the-shelf QP tools. After each $\hat{\boldsymbol{s}}_i$ is solved, we concatenate them together and obtain the updated graph matrix $\mathbf{S}$.

Update F While $\mathbf{S}$ and $\mathbf{W}$ are fixed, the optimization problem (4) can be stated as follows:

$$\begin{aligned}\min_{\mathbf{F}} \ & \|\mathbf{H}-\mathbf{F}\|_{\mathrm{F}}^2+\mu\sum_{j=1}^{m}\left\|\boldsymbol{f}_j-\sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{f}_i\right\|_2^2 \\ \text{s.t. } & \mathbf{F}\mathbf{1}_q=\mathbf{1}_m,\mathbf{0}_{m\times q}\le\mathbf{F}\le\mathbf{Y}, \end{aligned} \tag{8}$$

where $\mathbf{F}=[\boldsymbol{f}_1,\boldsymbol{f}_2,...,\boldsymbol{f}_m]^\top\in\mathbb{R}^{m\times q}$, and $\mathbf{H}\in\mathbb{R}^{m\times q}$ is denoted as the modeling output matrix. The first term of the above objective function guarantees the consistency between the model outputs and labeling confidences, and the second term considers the manifold assumption of label space. It is hard to optimize this function directly, thus we reformulate problem (8) so as to update $\mathbf{F}$.

Let $\tilde{\boldsymbol{f}}=vec(\mathbf{F})\in[0,1]^{mq}$ where $vec(\bullet)$ is the vectorization operator. Likewise, $\tilde{\boldsymbol{h}}=vec(\mathbf{H})\in\mathbb{R}^{mq}$ and $\tilde{\boldsymbol{y}}=vec(\mathbf{Y})\in\{0,1\}^{mq}$. We define a square matrix: $\mathbf{T}=2(\mathbf{I}_{\mathbf{m}\times\mathbf{m}}-\mathbf{S})(\mathbf{I}_{\mathbf{m}\times\mathbf{m}}-\mathbf{S})^\top$ where $\mathbf{I}_{m\times m}$ is an identity matrix with m rows and m columns. By denoting the cost function of problem (8) as $\mathcal{J}(\mathbf{F})$, then we have:

$$\begin{aligned}\mathcal{J}(\mathbf{F}) &= \sum_{i=1}^{m}\sum_{j=1}^{q}(h_{ij}-f_{ij})^2+\mu\sum_{j=1}^{m}\left\|\mathbf{F}^\top(\mathbf{I}_{m\times m})_{\cdot i}-\mathbf{F}^\top\mathbf{S}_{\cdot j}\right\|_2^2 \\ &= \sum_{i=1}^{m}\sum_{j=1}^{q}(h_{ij}^2-2h_{ij}f_{ij}+f_{ij}^2)+\mu\left\|(\mathbf{I}_{m\times m}-\mathbf{S})^\top\mathbf{F}\right\|_{\mathrm{F}}^2 \\ &= \sum_{i=1}^{m}\sum_{j=1}^{q}(-2h_{ij}f_{ij}+f_{ij}^2)+\frac{\mu}{2}tr(\mathbf{F}^\top\mathbf{T}\mathbf{F})+C \\ &= \frac{\mu}{2}\sum_{i=1}^{q}\mathbf{F}_{\cdot i}^\top\mathbf{T}\mathbf{F}_{\cdot i}+\sum_{i=1}^{q}\mathbf{F}_{\cdot i}^\top\mathbf{F}_{\cdot i}-2\sum_{i=1}^{q}\mathbf{H}_{\cdot i}^\top\mathbf{F}_{\cdot j}+C. \end{aligned} \tag{9}$$

Here C is a constant and equal to $\sum_{i=1}^{m}\sum_{j=1}^{q}h_{ij}$, while $\sum_{i=1}^{q}\mathbf{F}_{\cdot i}^\top\mathbf{T}\mathbf{F}_{\cdot i}$ is equal to $\tilde{\boldsymbol{f}}^\top\boldsymbol{\Lambda}\tilde{\boldsymbol{f}}$, where $\boldsymbol{\Lambda}\in\mathbb{R}^{mq\times mq}$ is defined as follows:

$$\boldsymbol{\Lambda}=\begin{bmatrix}\mathbf{T} & \mathbf{0}_{m\times m} & \cdots & \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m} & \mathbf{T} & \ddots & \vdots\\ \vdots & \ddots & \ddots & \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m} & \cdots & \mathbf{0}_{m\times m} & \mathbf{T}\end{bmatrix}. \tag{10}$$

Then minimizing the above function is equivalent to minimize the following function:

$$\begin{aligned}\tilde{\mathcal{J}}(\tilde{\boldsymbol{f}}) &= \frac{\mu}{2}\tilde{\boldsymbol{f}}^\top\boldsymbol{\Lambda}\tilde{\boldsymbol{f}}+\tilde{\boldsymbol{f}}^\top\tilde{\boldsymbol{f}}-2\tilde{\boldsymbol{h}}^\top\tilde{\boldsymbol{f}} \\ &= \frac{1}{2}\tilde{\boldsymbol{f}}^\top(\boldsymbol{\Lambda}+\frac{2}{\mu}\mathbf{I}_{mq\times mq})\tilde{\boldsymbol{f}}-\frac{2}{\mu}\tilde{\boldsymbol{h}}^\top\tilde{\boldsymbol{f}}. \end{aligned} \tag{11}$$

Thus the optimization problem (8) can be reformulated as follows:

$$\begin{aligned}\min_{\tilde{\boldsymbol{f}}} \ & \frac{1}{2}\tilde{\boldsymbol{f}}^\top\left(\boldsymbol{\Lambda}+\frac{2}{\mu}\mathbf{I}_{mq\times mq}\right)\tilde{\boldsymbol{f}}-\frac{2}{\mu}\tilde{\boldsymbol{h}}^\top\tilde{\boldsymbol{f}} \\ \text{s.t. } & \mathbf{0}_{mq}\le\tilde{\boldsymbol{f}}\le\tilde{\boldsymbol{y}},\ \textstyle\sum_{j=1,j\%m=i}^{mq}\tilde{\boldsymbol{f}}_j=1(\forall 0\le i\le m-1). \end{aligned} \tag{12}$$

It is obviously that the optimization problem (12) also corresponds a standard Quadratic Programming problem, which can be solved by off-the-shelf QP tools. The second constraint of problem (12) can be implemented in matrix form when using QP tools. Specifically, let $\mathbf{B}=[\mathbf{I}_{m\times m},\mathbf{I}_{m\times m},...,\mathbf{I}_{m\times m}]\in\{0,1\}^{m\times mq}$, then the constraint can be represented as $\mathbf{B}\tilde{\boldsymbol{f}}=\mathbf{1}_m$. Note that this QP problem have mq variables and $m(q+1)$ constraints, thus the computational complexity would be demanding when mq is large. Following [69] we employ alternating optimization strategy for large-scale data sets. For labeling confidence vector $\boldsymbol{f}_j$, it can be optimized by fixing all other labeling confidence vectors:

$$\begin{aligned}\min_{\boldsymbol{f}_j} \ & (\frac{t_{jj}}{2}+\frac{1}{\mu})\boldsymbol{f}_j^\top\boldsymbol{f}_j+\left(\textstyle\sum_{i=1,i\neq j}^{m}t_{ij}\boldsymbol{f}_i^\top-\frac{2}{\mu}\boldsymbol{h}_i^\top\right)\boldsymbol{f}_j \\ \text{s.t. } & \mathbf{0}_q\le\boldsymbol{f}_j\le\boldsymbol{y}_j,\ \boldsymbol{f}_j\mathbf{1}_q=1. \end{aligned} \tag{13}$$

Now the problem (12) is transformed into a series of QP subproblems with q variables and $q+1$ constraints which

TABLE 2
The pseudo-code of PL-AGGD

Input:

$\mathcal{D}$: the partially labeled training set $\{(\boldsymbol{x}_i, Y_i)|1 \leq i \leq m\}$ ($\boldsymbol{x}_i \in \mathcal{X}, Y_i \in \mathcal{Y}$, $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{0,1\}^q$)

k: the number of nearest neighbors used for weighted graph construction

μ, γ, λ: the trade-off parameters in objective function (4)

$\boldsymbol{x}$: the unseen instance to be predicted

Output:

y: the predicted label for instance $\boldsymbol{x}$

Process:

Calculate the kernel matrix $\mathbf{K}$=$[\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{m\times m}$ $(1 \leq i, j \leq m)$ over training examples;
Set the similarity graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$ with $\mathcal{V} = \{\boldsymbol{x}_i|0 \leq i \leq m\}$ and $\mathcal{E} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j)|\boldsymbol{x}_i \in KNN(\boldsymbol{x}_j), i \neq j\}$;
Initial the similarity matrix $\mathbf{S}$ by solving problem (1);
Initial the label confidence matrix $\mathbf{F}$ by solving problem (2);
repeat
  Update the matrix $\mathbf{A}$ and bias vector $\mathbf{b}$ according to (19);
  Calculate the model output matrix according to $\mathbf{H} = \frac{1}{2\lambda}\mathbf{K}\mathbf{A} + \mathbf{1}_m\mathbf{b}^\top$;
  for $j = 1$ **to** m **do**
    Calculate the vector $\hat{\boldsymbol{s}}_j \in \mathbb{R}^k$ by solving problem (7) with QP procedure and update the j-th column $\mathbf{S}_{\cdot j}$ of the similarity matrix $\mathbf{S}$;
  end for
  Update the label confidence matrix $\mathbf{F}$ by solving problem (12) or iteratively solving a series subproblems as per Equation (13);
until convergence or maximum number of iterations being reached
return the predicted label y according to Equation (20).

can be solved more efficiently.

Update W With the simple linear model $g(\boldsymbol{x}_i) = \mathbf{W}^\top\boldsymbol{x}_i$ while fixing $\mathbf{F}$ and $\mathbf{S}$, the problem (4) can be stated as follows:

$$\min_{\mathbf{W}} \sum_i \left\|\mathbf{W}^\top\boldsymbol{x}_i - \boldsymbol{f}_i\right\|_2^2 + \lambda\left\|\mathbf{W}\right\|_{\mathrm{F}}^2. \tag{14}$$

This regularized least squares problem is simple and can be easily solved by gradient descent method or closed-form solutions:

$$\mathbf{W} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}_{m\times m})^{-1}\mathbf{X}^\top\mathbf{F}. \tag{15}$$

Such linear model may not be able to deal with the complex nonlinear case, thus in the next subsection we adopt the kernel extension to achieve better performance.

Complexity Analysis Our optimization algorithm mainly contains several steps. Before alternative optimization, we need to find the k nearest neighbors for each sample in training data. The typical method constructs the data structure index in $O(dm\log m)$ (we use KD-Tree in our implementation) and then searches the k nearest neighbors for all training samples in $O(km\log m)$. Then, our method needs to iteratively solve three optimization problem w.r.t. $\mathbf{S}$, $\mathbf{F}$ and $\mathbf{W}$. To solve the QP problems of (7) and (12), we use the interior point method [62] which requires constant iterations and each iteration can be computed in $O(n^3)$ (n denotes the number of variables to be solved). Therefore, $\mathbf{S}$ and $\mathbf{F}$ can be solved in $O(mk^3)$ and $O(m^3q^3)$ respectively. When mq is large, we can solve the problem of (13) as an efficient alternative when updating $\mathbf{F}$, which requires the complexity of $O(mq^3)$. Furthermore, the corresponding complexity for updating $\mathbf{W}$ is $O(m^3)$. In summary, the overall complexity of our optimization algorithm is the sum of these operations mentioned above.

3.5 Kernel Extension

Extension I Let $\phi(\bullet) : \mathbb{R}^d \rightarrow \mathbb{R}^h$ denote the feature mapping that maps the feature space to some higher dimensional Hilbert space with h dimensions, then we can train kernel ridge regression model. By further employing the bias term $\mathbf{b} \in \mathbb{R}^q$ in predictive function, the predictive function with kernel extension can be represented as $g(\boldsymbol{x}_i) = \mathbf{W}^\top\phi(\boldsymbol{x}_i) + \mathbf{b}$. Then (14) can be reformulated as following problem when updating $\mathbf{W}$:

$$\min_{\mathbf{W}} \sum_i \left\|\boldsymbol{\epsilon}_i\right\|_2^2 + \lambda\left\|\mathbf{W}\right\|_{\mathrm{F}}^2 \quad \text{s.t. } \boldsymbol{f}_i = \mathbf{W}^\top\phi(\boldsymbol{x}_i) + \mathbf{b} - \boldsymbol{\epsilon}_i. \tag{16}$$

By defining a matrix $\mathbf{E} = [\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, ..., \boldsymbol{\epsilon}_3]^\top \in \mathbb{R}^{m\times q}$, the above problem can be re-written in the following matrix form:

$$\min_{\mathbf{W}} tr(\mathbf{E}^\top\mathbf{E}) + \lambda tr(\mathbf{W}^\top\mathbf{W}) \quad \text{s.t. } \mathbf{F} = \mathbf{\Phi}\mathbf{W} + \mathbf{1}_m\mathbf{b}^\top - \mathbf{E}, \tag{17}$$

where $\mathbf{\Phi} = [\phi(x_1), \phi(x_2), ..., \phi(x_m)]^\top$. The lagrangian function of this problem is:

$$\begin{aligned}&\mathcal{L}(\mathbf{W}, \mathbf{E}, \mathbf{A}) \\ &= tr(\mathbf{E}^\top\mathbf{E}) + \lambda tr(\mathbf{W}^\top\mathbf{W}) - tr(\mathbf{A}^\top(\mathbf{\Phi}\mathbf{W} + \mathbf{1}_m\mathbf{b}^\top - \mathbf{E} - \mathbf{F})),\end{aligned} \tag{18}$$

where $\mathbf{A} \in \mathbb{R}^{m\times q}$ stores the Lagrange multipliers. We know that the optimum solution of (16) need to satisfy the following conditions:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{W}} &= 2\lambda \mathbf{W} - \mathbf{\Phi}^\top \mathbf{A} = \mathbf{0}, \qquad \frac{\partial \mathcal{L}}{\partial \mathbf{E}} = 2\mathbf{E} + \mathbf{A} = \mathbf{0}, \\ \frac{\partial \mathcal{L}}{\partial \mathbf{A}} &= \mathbf{\Phi W} + \mathbf{1}_m \mathbf{b}^\top - \mathbf{E} - \mathbf{F} = \mathbf{0}.\end{aligned} \tag{19}$$

From these linear equations we can obtain:

$$\begin{aligned}&\mathbf{W} = \frac{\mathbf{\Phi}^\top \mathbf{A}}{2\lambda}, \quad \mathbf{A} = \left(\frac{1}{2\lambda}\mathbf{K} + \frac{1}{2}\mathbf{I}_{m\times m}\right)^{-1}\left(\mathbf{F} - \mathbf{1}_m \mathbf{b}^\top\right) \\ &\mathbf{b} = \left(\frac{\mathbf{dF}}{\mathbf{d1}_m}\right)^\top, \quad \mathbf{d} = \mathbf{1}_m^\top \left(\frac{1}{2\lambda}\mathbf{K} + \frac{1}{2}\mathbf{I}_{m\times m}\right)^{-1},\end{aligned} \tag{20}$$

where $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^\top$ with its element $k_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ based on the corresponding kernel function $\kappa(\cdot,\cdot)$. For PL-AGGD, Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2})$ is employed with σ being the average distance among each pair of training examples. Then, by incorporating the specified kernel function, the modeling output matrix is denoted by $\mathbf{H} = \frac{1}{2\lambda}\mathbf{KA} + \mathbf{1}_m \mathbf{b}^\top$. Furthermore, the predicted label of the test example $\boldsymbol{x}$ yielded by our approach is given as:

$$y^* = \arg\max_k \sum_{i=1}^{m} a_{ik}\kappa(\boldsymbol{x}, \boldsymbol{x}_i). \tag{21}$$

Extension II Besides the kernelized predictive model, we can further map data to high dimensional space when obtaining the graph weights $\mathbf{S}$. To search k nearest neighbors, we need to calculate the distance in high dimensional Hilbert space, this can be efficiently done as follows [42]:

$$\begin{aligned}&\|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\|_2 \\ &= \sqrt{(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))^\top (\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))} \\ &= \sqrt{\kappa(\boldsymbol{x}_i, \boldsymbol{x}_i) - 2\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + \kappa(\boldsymbol{x}_j, \boldsymbol{x}_j)} \\ &= \sqrt{k_{ii} - 2k_{ij} + k_{jj}}.\end{aligned} \tag{22}$$

Now, with kernel mapping $\phi(\bullet)$,(6) can be reformulated as:

$$\begin{aligned}\min_{\mathbf{S}_{\cdot j}} \ &\mu \left\| \boldsymbol{f}_j - \textstyle\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j)\in\mathcal{E}} s_{ij} \boldsymbol{f}_i \right\|_2^2 \\ &+ \gamma \left\| \phi(\boldsymbol{x}_j) - \textstyle\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j)\in\mathcal{E}} s_{ij} \phi(\boldsymbol{x}_i) \right\|_2^2 \\ \text{s.t.} \ &\mathbf{S}_{\cdot j}^\top \mathbf{1}_m = 1, \ \mathbf{0}_m \le \mathbf{S}_{\cdot j} \le \mathbf{N}_{\cdot j}.\end{aligned} \tag{23}$$

The second termof (23) can be restated as:

$$\left\| \phi(\boldsymbol{x}_j) - \textstyle\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j)\in\mathcal{E}} s_{ij} \phi(\boldsymbol{x}_i) \right\|_2^2 = \hat{\boldsymbol{s}}_j^\top \mathbf{G}_\phi^{x_j} \hat{\boldsymbol{s}}_j, \tag{24}$$

where $\mathbf{G}_\phi^{x_j} = \mathbf{D}_\phi^{x_j} (\mathbf{D}_\phi^{x_j})^\top \in \mathbb{R}^{k\times k}$ is the Gram matrix corresponding to the Hilbert feature space and $\mathbf{D}_\phi^{x_j} = [\phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_{\mathcal{N}_j(1)}), \phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_{\mathcal{N}_j(2)}), ..., \phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_{\mathcal{N}_j(k)})]^\top \in \mathbb{R}^{k\times m}$. Here, the (a,b)-th element of $\mathbf{G}_\phi^{x_j}$ can be efficiently calculated as:

$$\begin{aligned}\mathbf{G}_\phi^{x_j}(a,b) &= (\phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_{\mathcal{N}_j(a)}))^\top (\phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_{\mathcal{N}_j(b)})) \\ &= \kappa(\boldsymbol{x}_j, \boldsymbol{x}_j) - \kappa(\boldsymbol{x}_j, \boldsymbol{x}_{\mathcal{N}_j(a)}) \\ &\quad - \kappa(\boldsymbol{x}_j, \boldsymbol{x}_{\mathcal{N}_j(b)}) + \kappa(\boldsymbol{x}_{\mathcal{N}_j(a)}, \boldsymbol{x}_{\mathcal{N}_j(b)}) \\ &= k_{jj} - k_{j\mathcal{N}_j(a)} - k_{j\mathcal{N}_j(b)} + k_{\mathcal{N}_j(a)\mathcal{N}_j(b)}.\end{aligned} \tag{25}$$

Then, we can replace $\mathbf{G}^{x_j}$ with $\mathbf{G}_\phi^{x_j}$ in (7) and optimize a similar QP problem with k variables to update $\mathbf{S}$.

TABLE 3
The pseudo-code of PL-AGGD$^+$

Input:
$\mathcal{D}$, k, μ, γ, λ (same as in Table 2)
n: the number of selected instances in each query round
c: query interval
Output:
The learned model.
Process:
1-4: Follow the same procedure as in Table 2.
5: $t = 0$;
6: **repeat**
7: **if** $t\%\, c = 0$ **do**
8: Query n instance $\{\boldsymbol{x}_{a_1}, \boldsymbol{x}_{a_2}, ..., \boldsymbol{x}_{a_n}\}$;
9: Update the partial label matrix $\mathbf{Y}$;
10: **end if**
11: Sequentially update $\mathbf{A}$, $\mathbf{b}$, $\mathbf{S}$ and $\mathbf{F}$;
12: $t = t + 1$;
13: **until** maximum number of iterations being reached
14: **return** the learned model.

Table 2 summarizes the complete procedure of PL-AGGD. Our approach firstly initializes $\mathbf{S}$ and $\mathbf{F}$ (Steps 2-4) by adopting manifold assumption with initial graph weights. Then the alternative optimization strategy is adopted to learn adaptive graph weights to guide label disambiguation (Steps 5-12) and model training simultaneously. Finally, the unseen instance is classified based on the learned predictive model (Step 13).

3.6 Active Query Selection

In this subsection, we consider the particular case in which the learner is allowed to selectively query examples from ambiguously labeled data for manual disambiguation. We found that the proposed training algorithm is feasible to interact with oracle in active learning setting, since the instance query can be blended with the optimization of model and labeling confidences. Table 3 shows the derived algorithm PL-AGGD$^+$, in which the learner/oracle interaction occurs at the c-th, $2c$-th... optimization iterations.

By bringing the active query in, PL-AGGD$^+$ needs to learn from updated data set which contains both ambiguous instances and new disambiguated instances. Suppose the learner queries n instances $\{\boldsymbol{x}_{a_1}, \boldsymbol{x}_{a_2}, ..., \boldsymbol{x}_{a_n}\}$ and their labels are provided by the oracle. We only need to set the corresponding rows of $\mathbf{Y}$ to the correct one-hot label vectors. Also note that in our graph guided training algorithm, the explicit instance-label pair from the oracle can not only include a direct training signal for this instance in the model training, but also influence the labeling confidences of other instances when updating the labeling confidences in optimization procedure.

Now we turn our attention to strategies for actively selecting query instances for disambiguation. To derive more useful information from the oracle feedback, the algorithm tends to choose instance with high uncertainty. Here we

introduce two simple but effective criterion to measure the uncertainty of each instance.

Entropy-based Uncertainty (**EU**) The uncertainty of an instance can be quantified via the entropy of the labeling confidences. For the instance $\boldsymbol{x}_i$, denote $\hat{\boldsymbol{f}}_i$ as the disambiguated label vector from oracle, then the the entropy reduction is:

$$U_E(\boldsymbol{x}_i) = E(\boldsymbol{f}_i) - E(\hat{\boldsymbol{f}}_i), \tag{26}$$

where $E(\cdot)$ is the entropy function. In active partial label learning, the disambiguated label set is a one-hot code vector, and its entropy is always 0. Thus, given a labeling confidence vector $\boldsymbol{f}_i$, we can quantify its uncertainty via estimating the expected entropy reduction which is equal to the entropy of the labeling confidence distribution $\boldsymbol{f}_i$: $E(\boldsymbol{f}_i) = \sum_{l=1}^{q} f_{il} \cdot \log f_{il}$.

Margin-based Uncertainty (**MU**) Another way for measure the uncertainty of instances is from the maximum margin view. For a partially labeled instance, there is only one label among its candidate labels which is valid. Given instance $\boldsymbol{x}_i$, if the margin between its maximum labeling confidence and other labeling confidences is large, then it seems that this instance is easy for our learning approach to automatically disambiguate its candidate labels; otherwise it is hard for the learning approach to disambiguate it. Thus the margin between the maximum confidence of the label from candidate labels and other labeling confidences can be considered as a criterion for uncertainty. Formally, margin-based uncertainty is represented as follows:

$$U_M(\boldsymbol{x}_i) = 1 - f_{io} + \max_{j \neq o, y_j \in Y_i}(f_{ij}), \tag{27}$$

where $o = \arg\max_{1 \leq j \leq q} f_{ij}$ denotes the label index which has the maximum labeling confidence.

4 Experiments

In this section, two series of comparative experiments on controlled data sets as well as real-world data sets are conducted.[1] We firstly introduce the experimental setup. Secondly, We report the detailed experimental results with statistical performance comparisons.

4.1 Experimental Setup

Data sets For comprehensive performance evaluation, we compare our proposed approach with other approaches on both controlled UCI data sets and real-world partial label data sets. Characteristics of the controlled UCI data sets and real-world data sets are summarized in Table 4 and Table 7 respectively. We follow the widely-used controlling protocol in previous partial label learning research [8], [9], [32], and generate artificial partial label data sets under different configurations of controlling parameters p, r and ϵ based on the given multi-class UCI data sets. Firstly, we use p to control the proportion of examples which are ambiguously labeled. Secondly, r controls the number of false positive labels in the candidate label set (i.e. $|Y_i| = r + 1$) of each example, and in our setting the false positive labels in the training set are generated without replacement. Thirdly, ϵ denotes the ambiguity degree which controls the co-occurring probability of one coupling candidate label and the ground-truth label. We achieve this by firstly choosing at random for each label a dominant co-occurring label which is sampled with probability ϵ, and then the rest of the labels are sampled uniformly with probability $(1-\epsilon)/(L-2)$ (there is a single extra label per example).

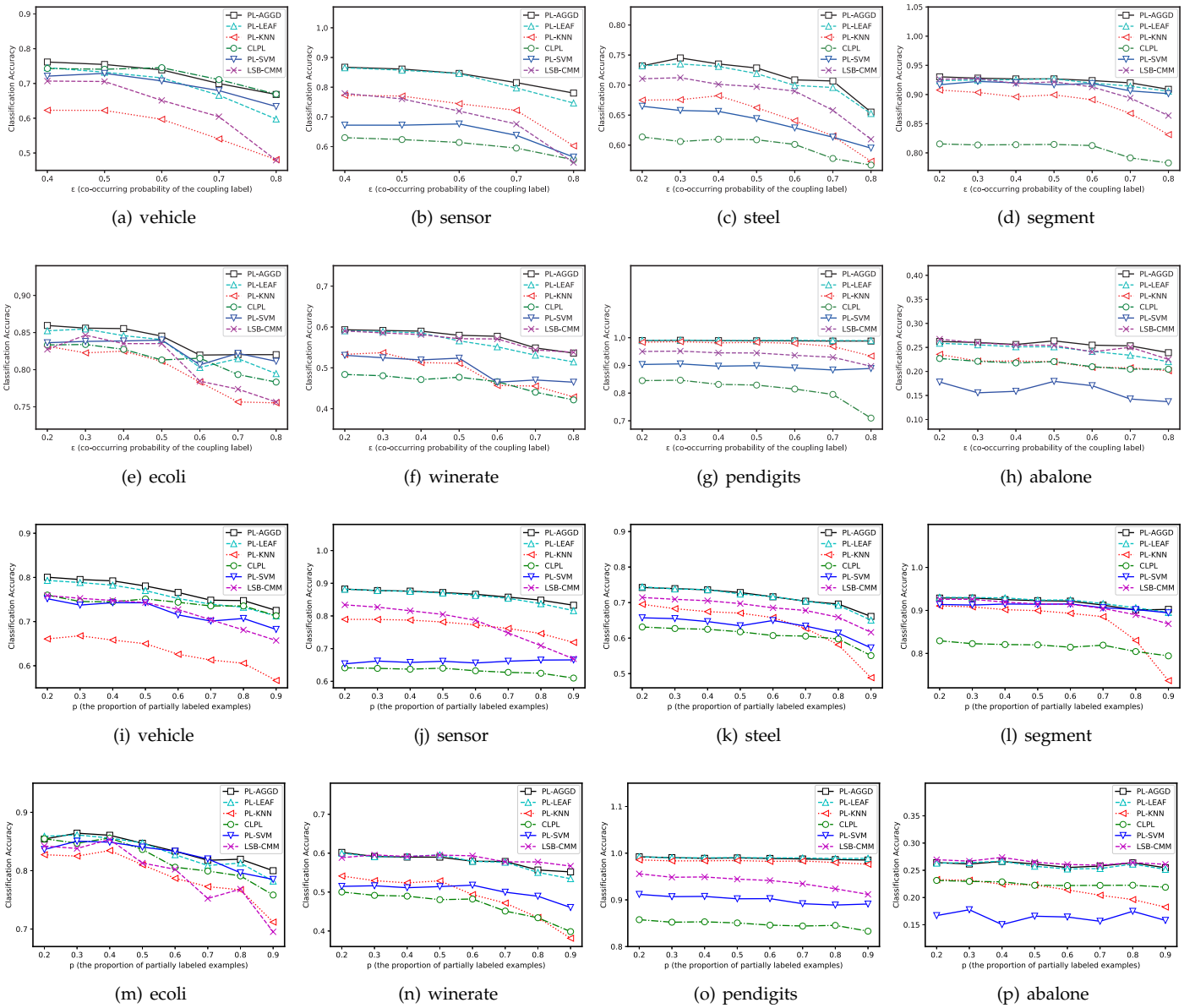
TABLE 4
Characteristics of the UCI data sets.

Data set	#examples	#features	#classes
vehicle	846	18	4
sensor	5456	24	4
steel	1941	27	7
segment	2310	18	7
ecoli	336	7	8
winerate	1599	11	10
pendigits	10992	16	10
abalone	4177	7	29

In addition, a number of real-world partially labeled data sets have been collected from several tasks and domains (see in Table 7) including `Lost` [10], `Soccer Player` [65], `Yahoo! News` [20] from automatic face naming, `MSRCv2` [32] from object classification, `FG-NET` [39] from facial age estimation, `BirdSong` [5] from bird song classification, `Malagasy` [16] and `Italian` [28] from POS tagging, and `Mirflickr` [24] from web image classification. For the automatic face naming task, faces cropped from an image are represented as instances while names extracted from the associated captions or subtitles are regarded as the corresponding candidate labels. For the task of object classification, image segmentations are regarded as instances while objects appearing within the same image are the corresponding candidate labels. For the task of facial age estimation, a human face is an instance while age numbers annotated by ten crowdsourced labelers as well as the ground-truth age are regarded as candidate labels. For the bird song classification, singing syllables of the birds are represented as instances while bird species jointly singing during a 10-seconds period are regarded as candidate labels. For web image classification, images are represented as instances while tags extracted from the web page are regarded as candidate labels. For POS tagging tasks, a given word with contextual information can be considered as an instance and all possible POS tags are regarded as the candidate labels. The average number of candidate labels (avg. #CLs) for each real-world partial label data set is also recorded in Table 7.

Compared Methods Five well-established partial label learning algorithms are employed for comparative studies, including two averaging-based partial label learning approaches PL-KNN [25] and CLPL [9], two identification-based approaches PL-SVM [38] and LSB-CMM [32], as well as feature-aware disambiguation approach PL-LEAF [69]. Each compared algorithm is implemented with the default hyper parameters setup suggested in respective literatures.

[1] All the real-world partial label data sets used in this paper and our code implementation of PL-AGGD are publicly-available at: http://palm.seu.edu.cn/zhangml/Resource.htm#partial_data and http://palm.seu.edu.cn/zhangml/files/PL-AGGD.rar



(a) vehicle (b) sensor (c) steel (d) segment

(e) ecoli (f) winerate (g) pendigits (h) abalone

(i) vehicle (j) sensor (k) steel (l) segment

(m) ecoli (n) winerate (o) pendigits (p) abalone

Fig. 4. The first two rows illustrate the classification accuracy of each comparing algorithm with varying ϵ (co-occurring probability of the coupling label) and fixed r and p ($r = 1, p = 1$).The last two rows illustrate the classification accuracy of each comparing algorithm with varying p (the proportion of examples which are partially labeled) and fixed r and ϵ ($r = 2$ for `vehicle` and `sensor`, $r = 5$ for others, $\epsilon = \frac{1}{r}$).

4.2 Main Experimental Results

In this subsection, we present the comparison results on both controlled and real-world data sets. The parameters of the proposed approach PL-AGGD are set as $k = 10$, $T = 20$, $\lambda = 1$, $\mu = 1$ and $\gamma = 0.05$. We perform ten runs of 50%/50% random train/test splits on each artificial as well as real-world partial label data set, and the mean accuracies (with standard deviations) are recorded for all algorithms. The parameter sensitivity analysis of PL-AGGD is conducted in Subsection 4.3. We use kernel extension I in our main experiments, and the comparison results with linear model and kernel extension II will be presented in ablation study in Subsection 4.5.

Controlled UCI Data Sets The classification results on the controlled UCI data sets are reported in Fig. 4 and Table 5. The top two rows of Fig. 4 illustrate the classification accuracy of each comparing algorithm with varying label co-occurring probability ϵ from 0.1 to 0.8 with step size 0.1. For any ground-truth label $y \in Y$, one extra label $y' \neq y$ is designated as the coupling label which co-occurs with y in the candidate label set with probability ϵ ($r = 1$). The last two rows illustrate the classification accuracy of each comparing algorithm with varying proportion p of examples which are partially labeled. For each example, there will be r other labels randomly chosen to co-occur with the ground-truth label. Note that for `vehicle` and `sensor`, there are only 4 labels in each of these two data sets, and the minimum ambiguity degree of all label pairs is $1/3$, therefore we vary ϵ from 0.4 to 0.8 in Fig. 4a and 4b. In the experiments with varying p, we set $r = 2$ for `vehicle` and `sensor` and $r = 5$ for other data sets.

Furthermore, we vary the ambiguity size, i.e. the number

TABLE 5
Classification accuracy (mean±std) of each comparing algorithm on the controlled UCI data sets with varying r (false positive candidate label). In addition, •/○ indicates whether the performance of PL-AGGD is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significance level).

Data set	r	PL-AGGD	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
vehicle	1	0.770±0.025	0.641±0.017•	0.756±0.025•	0.727±0.027•	0.720±0.022•	0.753±0.024•
	2	0.686±0.028	0.512±0.018•	0.708±0.015•	0.649±0.032•	0.604±0.025•	0.670±0.033•
sensor	1	0.866±0.004	0.771±0.006•	0.633±0.010•	0.682±0.010•	0.775±0.010•	0.865±0.005
	2	0.811±0.008	0.692±0.015•	0.605±0.013•	0.650±0.023•	0.649±0.006•	0.788±0.006•
steel	1	0.733±0.019	0.677±0.009•	0.614±0.016•	0.660±0.013•	0.708±0.013•	0.730±0.017
	3	0.703±0.014	0.605±0.011•	0.579±0.026•	0.612±0.026•	0.656±0.009•	0.693±0.016•
	5	0.580±0.028	0.329±0.024•	0.501±0.038•	0.474±0.056•	0.528±0.029•	0.550±0.034•
segment	1	0.927±0.006	0.907±0.009•	0.811±0.008•	0.915±0.010•	0.925±0.008	0.929±0.005
	3	0.908±0.007	0.878±0.000•	0.805±0.016•	0.916±0.004○	0.896±0.011•	0.912±0.010○
	5	0.862±0.019	0.512±0.032•	0.780±0.016•	0.816±0.066•	0.765±0.026•	0.847±0.020•
ecoli	1	0.837±0.024	0.804±0.033•	0.815±0.026•	0.828±0.033	0.825±0.029	0.833±0.024
	3	0.828±0.028	0.796±0.027•	0.805±0.034•	0.816±0.026	0.767±0.027•	0.833±0.031
	5	0.727±0.063	0.633±0.068•	0.697±0.065•	0.667±0.086•	0.622±0.070•	0.716±0.067•
winerate	1	0.588±0.014	0.533±0.018•	0.469±0.016•	0.512±0.018•	0.581±0.012	0.587±0.012
	3	0.574±0.018	0.452±0.029•	0.438±0.028•	0.502±0.021•	0.581±0.015○	0.549±0.018•
	5	0.515±0.032	0.315±0.024•	0.377±0.022•	0.423±0.042•	0.547±0.018○	0.469±0.037•
pendigits	1	0.990±0.001	0.984±0.001•	0.849±0.002•	0.905±0.004•	0.951±0.003•	0.991±0.000○
	3	0.990±0.001	0.983±0.001•	0.832±0.005•	0.889±0.007•	0.919±0.009•	0.990±0.001
	5	0.989±0.001	0.966±0.002•	0.830±0.006•	0.857±0.055•	0.882±0.004•	0.988±0.001
abalone	1	0.260±0.008	0.230±0.006•	0.227±0.008•	0.158±0.020•	0.265±0.014○	0.259±0.009
	4	0.253±0.005	0.187±0.004•	0.219±0.010•	0.149±0.030•	0.256±0.006	0.248±0.005•
	7	0.248±0.010	0.150±0.011•	0.204±0.006•	0.156±0.033•	0.239±0.009•	0.238±0.008•
	10	0.241±0.009	0.122±0.008•	0.190±0.007•	0.184±0.019•	0.230±0.012•	0.227±0.008•

TABLE 6
Win/tie/loss counts on the classification performance of PL-AGGD against each comparing algorithm on controlled UCI data sets.

	PL-AGGD **against**				
	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
[varying ϵ]	52/0/0	47/5/0	45/7/0	38/13/1	29/20/3
[varying p]	64/0/0	59/5/0	57/7/0	46/14/4	20/36/8
[varying r]	23/0/0	23/0/0	10/2/1	16/4/3	13/8/2
In total	**139/0/0**	**129/10/0**	**122/16/1**	**100/31/8**	**62/64/13**

of extra labels associated with each example (see in Table 5). For each example, along with the ground-truth label, r extra class labels will be randomly picked up to constitute the candidate label set.

As shown in Fig. 4 and Table 5, PL-AGGD achieves superior or competitive performance against the comparing algorithms. Table 6 summarizes the win/tie/loss counts between PL-AGGD and each comparing algorithm. Out of the 139 statistical tests on all 8 UCI data sets, it is shown that:

- Comparing to averaging-based disambiguation approaches, PL-AGGD achieves superior performance against PL-KNN and CLPL in 100% and 93% cases respectively.
- Comparing to identification-based disambiguation approaches, PL-AGGD achieves superior performance against PL-SVM and LSB-CMM in 88% and 72% cases respectively. Furthermore, the performance of PL-AGGD is inferior to PL-SVM in only 1 case and has been outperformed by LSB-CMM in only 8 cases.
- Comparing to existing feature-aware approach PL-LEAF, PL-AGGD achieves superior performance in 45% cases and is outperformed by PL-LEAF in only 9% cases. Furthermore, from Fig. 4 and Table 5 we can observe that PL-AGGD is more robust when class ambiguity degree ϵ or ambiguity size r is large.

Real-World Data Sets The classification accuracy of each comparing algorithm on the real-world partial label data sets is reported in Table 8. Note that the average size of candidate label set (avg. #CLs) of the data set FG-NET is quite large, which could cause the extremely low classification accuracy of all compared algorithms on the test data set. The common evaluation criteria on this task is mean average error (MAE) between the predicted age and the

TABLE 7
Characteristics of the real-world data sets.

Data set	#examples	#features	#classes	avg. #CLs	Task Domain
FG-NET	1002	262	78	7.48	*facial age estimation* [39]
Lost	1122	108	16	2.23	*automatic face naming* [10]
MSRCv2	1758	48	23	3.16	*object classification* [32]
Mirflickr	2780	1536	14	2.76	*web image classification*[24]
Birdsong	4998	38	13	2.18	*bird song classification* [5]
Malagasy	5303	384	44	8.35	*POS Tagging* [16]
Soccer Player	17472	279	171	2.09	*automatic face naming* [65]
Italian	21878	519	90	1.6	*POS Tagging* [28]
Yahoo! News	22991	163	219	1.91	*automatic face naming* [20]

TABLE 8
Classification accuracy (mean±std) of each comparing algorithm on the real-world partial label data sets. In addition, •/◦ indicates whether the performance of PL-AGGD is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significance level).

Data set	PL-AGGD	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
FG-NET(MAE3)	0.427±0.012	0.278±0.019•	0.320±0.038•	0.347±0.018•	0.353±0.016•	0.407±0.022•
FG-NET(MAE5)	0.566±0.018	0.425±0.019•	0.451±0.043•	0.483±0.019•	0.502±0.021•	0.544±0.028•
Lost	0.711±0.010	0.308±0.025•	0.662±0.022•	0.682±0.031•	0.594±0.021•	0.678±0.016•
MSRCv2	0.464±0.014	0.403±0.017•	0.399±0.016•	0.430±0.014•	0.404±0.016•	0.470±0.018
Mirflickr	0.652±0.016	0.488±0.022•	0.660±0.014	0.607±0.032•	0.580±0.030•	0.634±0.014•
BirdSong	0.717±0.008	0.627±0.006•	0.624±0.009•	0.663±0.018•	0.683±0.003•	0.707±0.008•
Malagasy	0.651±0.020	0.577±0.007•	0.602±0.009•	0.608±0.009•	0.551±0.009•	0.622±0.018•
Soccer Player	0.524±0.005	0.494±0.004•	0.348±0.005•	0.485±0.004•	0.504±0.002•	0.516±0.003•
Italian	0.698±0.013	0.510±0.007•	0.724±0.011◦	0.596±0.024•	0.656±0.002•	0.675±0.009•
Yahoo! News	0.613±0.005	0.448±0.005•	0.459±0.005•	0.609±0.006	0.585±0.005•	0.607±0.002•

ground-truth age. In Table 8, for better evaluation of this facial age estimation task, the comparison experiment of data set `FG-NET` are evaluated on MAE3 and MAE5, i.e. test examples are considered to be correctly classified if the difference between the predicted age and the ground-truth age is no more than 3 years and 5 years.

The classification results of real-world partial label learning tasks are shown in Table 8. It is obvious that PL-AGGD outperforms all the counterpart algorithms on these real-world tasks expect for CLPL on `Mirflickr`, PL-SVM on `Yahoo! News` and PL-LEAF on `MSRCv2`. Furthermore, our approach is only outperformed by CLPL on the `Italian` data set.

4.3 Further Analysis

Transductive Accuracy In addition to the inductive performance, it is also interesting to evaluate the transductive performance of each algorithm on the training examples [9], [69][2]. Transductive performance reflects the disambiguation ability of each partial label learning algorithm. For PL-AGGD and PL-LEAF, the generated labeling confidence vector $\boldsymbol{f}_i$ can be used to determine the ground-truth label of a training example x_i as $\hat{y}_i = \arg\max_{y_k \in Y_i} \boldsymbol{f}_{ik}$. For other approach, the ground-truth label can be determined by consulting the candidate label set Y_i, i.e. set $\hat{y}_i \in Y_i$ with largest modeling output. Accordingly, the transductive accuracy of each comparing algorithm on real-world data sets is reported in Table 10. Out of the 50 statistical tests (10 data sets × 5 comparing algorithm), it is shown that:

- Comparing to averaging-based disambiguation approaches, PL-AGGD is only outperformed by CLPL on the `Lost` data set. In terms of the rest statistical tests, the performance of PL-AGGD is superior or at least comparable to PL-KNN and CLPL.
- Comparing to identification-based disambiguation approaches, PL-AGGD is outperformed by PL-SVM on the `Yahoo! News` data set. In terms of the rest statistical tests, the performance of PL-AGGD is superior or

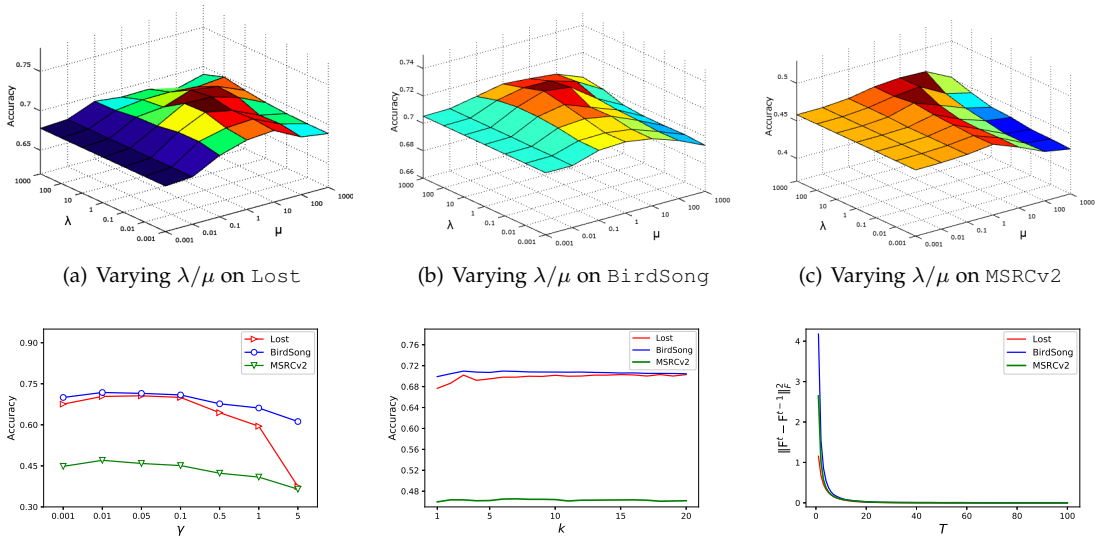
TABLE 9
Win/tie/loss counts on the transductive performance of PL-AGGD against each comparing algorithm on controlled UCI data sets.

	PL-AGGD **against**				
	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
[varying ϵ]	48/4/0	48/2/2	43/7/2	26/11/15	39/13/0
[varying p]	52/11/1	62/2/0	58/6/0	33/12/19	46/10/8
[varying r]	23/0/0	20/2/1	19/4/0	11/4/8	17/4/2
In total	**123/15/1**	**130/6/3**	**120/17/2**	**70/27/42**	**102/27/10**

[2]Note that induction refers to learning a predictive function that can be applied to any novel instance, while transduction is only concerned with the disambiguation performance on training data.

TABLE 10
Transductive accuracy (mean±std) of each comparing algorithm on the real-world partial label data sets. In addition, •/◦ indicates whether the performance of PL-AGGD is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significance level).

Data set	PL-AGGD	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
FG-NET(MAE3)	0.565±0.013	0.545±0.023•	0.547±0.014•	0.531±0.029•	0.530±0.015•	0.546±0.015•
FG-NET(MAE5)	0.701±0.014	0.702±0.022	0.694±0.021	0.670±0.027•	0.674±0.013•	0.681±0.019•
Lost	0.830±0.015	0.586±0.024•	0.848±0.019◦	0.832±0.027	0.748±0.023•	0.776±0.019•
MSRCv2	0.630±0.009	0.567±0.020•	0.604±0.025•	0.619±0.026•	0.592±0.023•	0.628±0.008
Mirflickr	0.697±0.010	0.567±0.015•	0.706±0.013	0.669±0.027•	0.632±0.013•	0.665±0.014•
BirdSong	0.826±0.012	0.751±0.014•	0.758±0.015•	0.830±0.014	0.812±0.015	0.782±0.016•
Malagasy	0.806±0.020	0.797±0.008	0.724±0.013•	0.790±0.015	0.710±0.006•	0.767±0.025•
Soccer Player	0.715±0.005	0.652±0.004•	0.626±0.003•	0.710±0.006	0.684±0.003•	0.701±0.004•
Italian	0.831±0.011	0.686±0.005•	0.684±0.009•	0.793±0.018•	0.814±0.004•	0.802±0.005•
Yahoo! News	0.837±0.003	0.700±0.003•	0.746±0.008•	0.842±0.002◦	0.830±0.003•	0.820±0.003•



(a) Varying λ/μ on Lost (b) Varying λ/μ on BirdSong (c) Varying λ/μ on MSRCv2

(d) Varying γ (e) Varying k (f) Convergence rate

Fig. 5. Parameter sensitivity analysis for PL-AGGD. (a) Classification accuracy of PL-AGGD on Lost by varying λ and μ; (b) Classification accuracy of PL-AGGD on BirdSong by varying λ and μ; (c) Classification accuracy of PL-AGGD on MSRCv2 by varying λ and μ; (d) Classification accuracy of PL-AGGD on Lost, BirdSong and MSRCv2 by varying γ; (e) Classification accuracy of PL-AGGD on Lost, BirdSong and MSRCv2 by varying k; (f) Convergence curves of the labeling scores on Lost, BirdSong and MSRCv2.

at least comparable to PL-SVM and LSB-CMM.

- Comparing to existing feature-aware approach, the disambiguation performance of PL-AGGD is superior or at least comparable to PL-LEAF on all data sets.

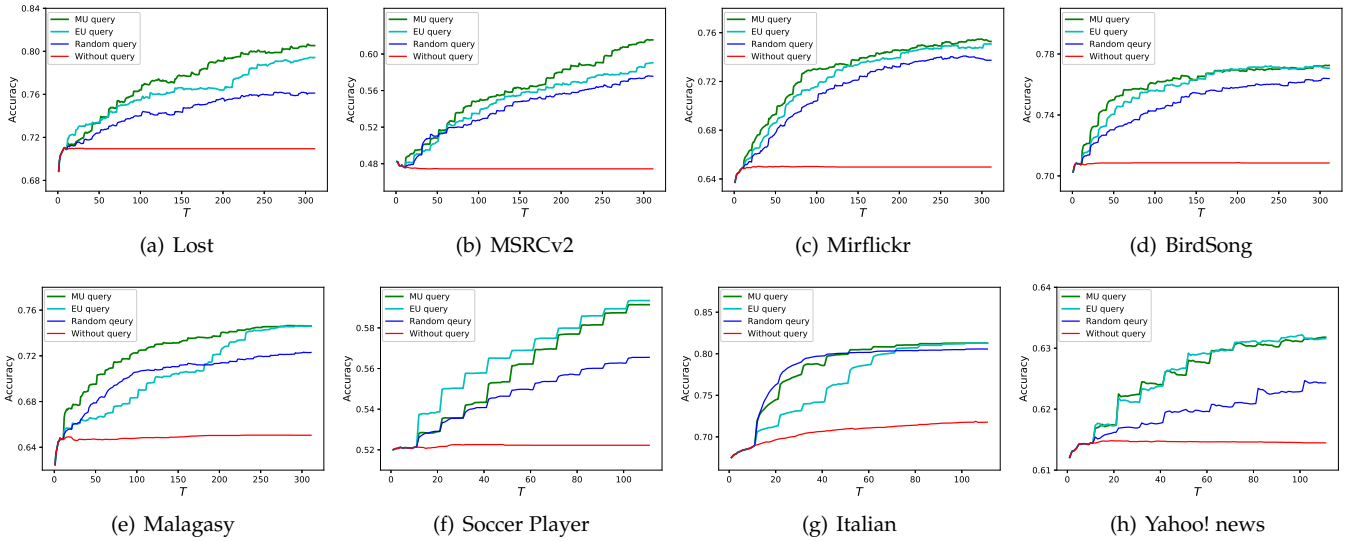
We also conduct the comparison experiments of transductive performance on controlled UCI data sets. Table 9 summarizes the win/tie/loss counts of transductive accuracy between PL-AGGD and each comparing algorithm of the 139 statistical tests on all 8 UCI data sets.[3]

Parameter Sensitivity The sensitivity of PL-AGGD with respect to its four parameters λ, μ, γ and k is also studied. The performance of PL-AGGD under different parameter configurations on Lost, BirdSong and Mirflickr is illustrated in Fig. 5. As shown in Fig. 5(a)-(c), when μ and λ are too small or too large, PL-AGGD gives a poor performance. In practice, we suggest users to choose λ and μ around 1 for reliable usage. Trade-off parameter γ aims to control the model complexity. The classification accuracy curves of varying γ (Fig. 5(d)) obviously coincides with the fact that it is important to balance between over-fitting and under-fitting. In practice, the value of γ can be manually searched prior to model training. The performance sensitivity of different k is shown in Fig. 5(e), in which the accuracy curves state that the performance of PL-AGGD is not sensitive w.r.t. k especially when k is larger than 5. Fig. 5(f) shows the convergence curves of PL-AGGD by using difference between the labeling confidence matrices $\mathbf{F}$ (measured by L_2 norm $\|\mathbf{F}^{(t)}-\mathbf{F}^{(t-1)}\|_{\mathrm{F}}^2$) of two adjacent iterations. It is shown that the labeling confidences converge with increasing number of iterations ($\mathbf{F}$ becomes convergent when T reaches 20).

[3]For brevity, detailed comparison results are provided in supplementary file.

4.4 Active Learning Performance

Now we report the active learning performance of PL-AGGD$^+$. We compare 3 active query strategies, Margin-



(a) Lost (b) MSRCv2 (c) Mirflickr (d) BirdSong

(e) Malagasy (f) Soccer Player (g) Italian (h) Yahoo! news

Fig. 6. Active learning performance of each comparing query strategy. We compare 3 active query strategies, *Margin-based Uncertainty* query strategy, *Entropy-based Uncertainty* query strategy and *Random* query strategy, as well as vanilla PL-AGGD method *without* active query.

TABLE 11
Win counts of each query strategy in cases where $3\times1\%$, $3\times2\%$, ..., $3\times10\%$ of training examples are selected for manual disambiguation.

Data set	**MU**	**EU**	Random
Lost	9	1	0
MSRCv2	9	0	1
Mirflickr	10	0	0
Birdsong	6	4	0
Malagasy	10	0	0
Soccer Player	0	10	0
Italian	6	0	4
Yahoo! News	4	6	0
Total	**54**	**21**	**5**

based Uncertainty query strategy, Entropy-based Uncertainty query strategy and Random query strategy, as well as vanilla PL-AGGD approach *without* active query. Note that we fix $c = 10$ in PL-AGGD$^+$ for all data sets, and set $n = m/100$ for first 6 data sets while $n = 3 \times m/100$ for last 3 data sets. Thus there could be at most 30% of training examples queried for disambiguation for all data sets. The average accuracy curves of different compared strategies with queried instances increasing are plotted in Fig. 6. We can see that active learning strategies achieved better performance than vanilla PL-AGGD on all these 8 data set. Furthermore, our proposed strategies **MU** and **EU** outperform Random query strategy in most cases. We also conduct the comparison results when $3\times1\%, 3\times2\%, ..., 3\times10\%$ of the number of queries is reached in Table 11. The statistic results in Table 11 show that **MU** and **EU** based strategies are better than random query in 94% cases, and **MU** based query strategy achieved superior performance than **EU** based strategy in more cases.

Note that the comparison results on FG-NET are not presented in Table 11 and Fig. 6, because in facial age estimation tasks the ambiguous labels (which are near by the ground-truth labels) can also provide useful information for training. In active learning setting, the manual disambiguation, which results in a one-hot label vector for each instance, will completely erase the information of these partial labels. On the contrary, the proposed automatic disambiguation, which results in a label distribution for each instance, will preserve the information of partial labels.

4.5 Ablation Study

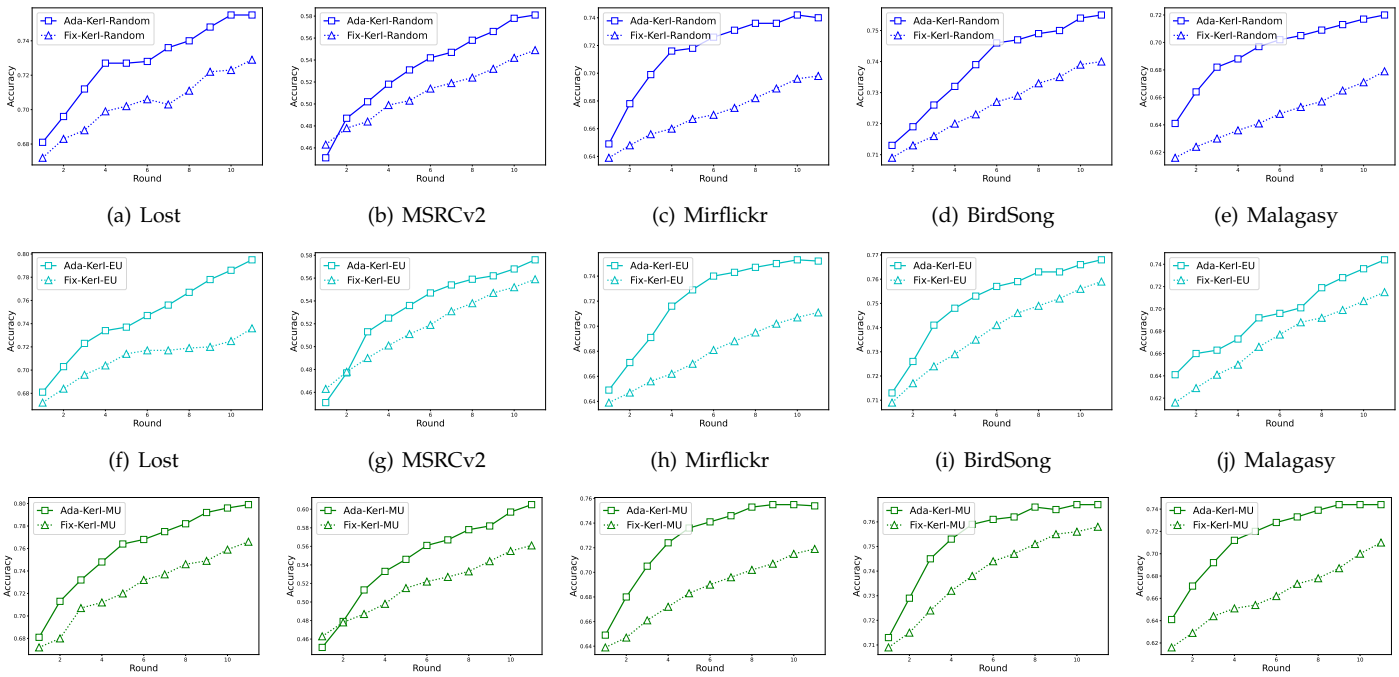
In this subsection, the ablation study on several variants of the proposed method is further conducted to analyze the essential building blocks. As we discussed in methodology section, our method uses a unified objective to learn the similarity graph weights, labeling confidences and model parameters simultaneously. To show the effectiveness of this adaptive-graph-based framework, we compare our method with a variant which separately learn each of these terms (denoted as Fix-KerI). Although the comparison method PL-LEAF also uses this fixed-graph-based strategy, it adopts a more complicate predictive model in training and testing phase. Thus, here we replace its predictive model with the kernelized multi-output model as presented in Subsection 3.5 for fair comparison. As the above presented experimental results are based kernel extension I, two variants named Ada-Linear and Ada-KerII are investigated which use linear model and kernel extension II respectively. In addition, here we also compare our method with two existing fixed-graph methods, i.e., GM-PLL [36] and LALO [13]. We also compare our method with the fixed-graph-based method in active learning setting. As is shown in Table 12 [4]:

- Adaptive-graph-based strategy clearly better than fixed-graph-based strategy. For example, Ada-KerI achieves superior performance in 27 cases out of the total 30 cases than Fix-KerI. Our methods also outperform GM-PLL and LALO in most cases.
- Kernel extensions outperform the linear model in most cases. On some particular data sets, the predictive accuracies of linear model are lower than those of kernel models with quite large margins.

[4]Here the results of our method are slightly different with those in Table 5 and Table 8 due to different random seeds.

TABLE 12
Ablation study of PL-AGGD with several variants on both the controlled UCI data sets and real-world partial label data sets. The best accuracies (mean±std) among 2-5 columns (for ordinary learning setting) and 6-7 columns (for active learning setting) are boldfaced respectively.

Data set	Fix-KerI	Ada-KerI (gain)	Ada-KerII	Ada-Linear	GM-PLL	LALO	Fix-KerI-**MU**	Ada-KerI-**MU** (gain)
vehicle (r=1)	0.757±0.009	**0.771±0.015** (0.014)	**0.771±0.017**	0.753±0.008	0.623±0.020	0.768±0.021	0.795±0.009	**0.813±0.012** (0.018)
vehicle (r=2)	0.686±0.023	0.703±0.022 (0.017)	0.694±0.023	**0.705±0.027**	0.482±0.035	0.697±0.029	0.777±0.016	**0.797±0.019** (0.020)
steel (r=3)	0.677±0.015	**0.698±0.013** (0.021)	**0.698±0.013**	0.606±0.013	0.549±0.009	0.695±0.010	0.727±0.011	**0.737±0.004** (0.010)
steel (r=5)	0.553±0.032	0.572±0.022 (0.019)	**0.576±0.020**	0.523±0.024	0.414±0.051	**0.576±0.023**	0.679±0.021	**0.711±0.013** (0.033)
segment (r=3)	**0.912±0.011**	0.906±0.008 (-0.006)	0.906±0.007	0.818±0.010	0.669±0.019	0.903±0.008	0.934±0.008	**0.937±0.008** (0.003)
segment (r=5)	0.861±0.013	**0.864±0.017** (0.003)	0.856±0.019	0.815±0.009	0.402±0.045	0.861±0.021	0.927±0.007	**0.936±0.005** (0.009)
ecoli (r=3)	0.815±0.028	0.818±0.020 (0.003)	**0.823±0.016**	0.794±0.039	0.741±0.026	0.819±0.020	0.842±0.019	**0.852±0.013** (0.010)
ecoli (r=5)	**0.765±0.041**	0.761±0.045 (-0.004)	0.756±0.054	0.756±0.045	0.699±0.033	0.760±0.042	0.833±0.021	**0.844±0.020** (0.011)
winerate (r=3)	0.553±0.012	0.572±0.016 (0.019)	**0.575±0.017**	0.409±0.034	0.447±0.025	0.570±0.016	0.582±0.006	**0.592±0.013** (0.010)
winerate (r=5)	0.459±0.015	0.518±0.017 (0.059)	**0.520±0.021**	0.342±0.026	0.397±0.032	**0.520±0.027**	0.552±0.010	**0.591±0.006** (0.039)
Lost	0.672±0.029	**0.681±0.026** (0.009)	**0.681±0.025**	0.698±0.027	0.654±0.024	0.667±0.032	0.766±0.013	**0.799±0.010** (0.033)
MSRCv2	**0.463±0.020**	0.451±0.025 (-0.012)	0.442±0.030	0.398±0.018	0.331±0.013	0.449±0.023	0.561±0.019	**0.605±0.021** (0.044)
Mirflickr	0.639±0.012	**0.649±0.014** (0.010)	0.625±0.013	0.453±0.019	0.422±0.009	0.638±0.014	0.719±0.014	**0.754±0.008** (0.035)
Birdsong	0.709±0.002	**0.713±0.006** (0.004)	0.710±0.004	0.631±0.008	0.556±0.010	0.701±0.009	0.758±0.008	**0.767±0.010** (0.009)
Malagasy	0.616±0.011	**0.641±0.009** (0.025)	0.557±0.042	0.572±0.033	0.228±0.011	0.635±0.009	0.710±0.006	**0.744±0.009** (0.034)



(a) Lost (b) MSRCv2 (c) Mirflickr (d) BirdSong (e) Malagasy

(f) Lost (g) MSRCv2 (h) Mirflickr (i) BirdSong (j) Malagasy

(k) Lost (l) MSRCv2 (m) Mirflickr (n) BirdSong (o) Malagasy

Fig. 7. The accuracy curves of our method AdaKerI and FixKerI in active learning setting with varied population of selectively queried instances. Top: *Random* query strategy. Middle: *Entropy-Based Uncertainty* query strategy. Bottom: *Margin-based Uncertainty* query strategy.

- When we are allowed to query examples for manual disambiguation, the superiority of our method is more obvious. We present the accuracy gains of our method compared with fixed-graph-based method in both ordinary and active learning settings. The average accuracy gain in ordinary setting is 0.012 and that in active setting is 0.021.

In Fig. 7, we plot the accuracy curves of our method and fix-graph-based method with varied population of selectively queried instances. We can see that our method achieves more rapid performance increasing with the help of manual disambiguation for all query strategies. This implies that our method can obtain good performance in active learning setting with only a few annotation times.

5 Conclusion

In this paper, a novel approach based on adaptive graph guided disambiguation is proposed which is an extended version to our earlier research [47] on feature-aware partial label learning. The proposed approach PL-AGGD learns from ambiguously labeled examples by performing label disambiguation and predictive model training simultaneously. Extensive comparative studies on controlled and real-world data sets clearly validate that adaptive graph guided disambiguation could be more robust to deal with partial label learning. Moreover, this paper considers the particular case in which the learner is allowed to selectively query examples from ambiguously labeled data for manual disambiguation and thus extends the proposed approach under active learning setting.

References

[1] J. Amores, "Multiple instance classification: Review, taxonomy and comparative study," *Artificial intelligence*, vol. 201, pp. 81–105, 2013.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Advances in Neural Information Processing Systems*, 2003, pp. 577–584.

[3] J. Baldridge and M. Osborne, "Active learning and the total cost of annotation," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 9–16.

[4] M. Bouguelia, S. Nowaczyk, K. C. Santosh, and A. Verikas, "Agreeing to disagree: active learning with noisy labels without crowdsourcing," *International Journal of Machine Learning and Cybernetics*, vol. 9, pp. 1307–1319, 2018.

[5] F. Briggs, X. Z. Fern, and R. Raich, "Rank-loss support instance machines for miml instance annotation," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 534–542.

[6] J. Chai, I. W. Tsang, and W. Chen, "Large margin partial label machine," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2019.

[7] C.-H. Chen, V. M. Patel, and R. Chellappa, "Learning from ambiguously labeled face images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 7, pp. 1653–1667, 2018.

[8] Y.-C. Chen, V. M. Patel, R. Chellappa, and P. J. Phillips, "Ambiguously labeled learning using dictionaries," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2076–2088, 2014.

[9] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *Journal of Machine Learning Research*, vol. 12, no. 5, pp. 1501–1536, 2011.

[10] T. Cour, B. Sapp, C. Jordan, and B. Taskar, "Learning from ambiguously labeled images," in *Proceedings of the 20th IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 919–926.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the 20th IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[12] M. Fang, J. Yin, and D. Tao, "Active learning for crowdsourcing using knowledge transfer," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, pp. 1809–1815.

[13] L. Feng and B. An, "Leveraging latent label distributions for partial label learning." in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 2107–2113.

[14] L. Feng and B. An, "Partial label learning by semantic difference maximization," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 2294–2300.

[15] L. Feng and B. An, "Partial label learning with self-guided retraining," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019, pp. 3542–3549.

[16] D. Garrette and J. Baldridge, "Learning a part-of-speech tagger from two hours of annotation," in *Proceedings of the 13th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 138–147.

[17] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, p. 52, 2015.

[18] C. Gong, T. Liu, Y. Tang, J. Yang, J. Yang, and D. Tao, "A regularization approach for instance-based superset label learning," *IEEE Transactions on Cybernetics*, vol. 48, no. 3, pp. 967–978, 2018.

[19] J. Gonsior, M. Thiele, and W. Lehner, "Weakal: Combining active learning and weak supervision," in *Proceedings of the 23rd International Conference on Discovery Science*, vol. 12323, 2020, pp. 34–49.

[20] M. Guillaumin, J. Verbeek, and C. Schmid, "Multiple instance metric learning from automatically labeled bags of faces," in *Proceedings of the 11th European Conference on Computer Vision*, 2010, pp. 634–647.

[21] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Advances in Neural Information Processing Systems*, 2018, pp. 8527–8537.

[22] P. Hu, Z. C. Lipton, A. Anandkumar, and D. Ramanan, "Active learning with partial feedback," in *the 7th International Conference on Learning Representations*, 2019.

[23] S. Huang, N. Gao, and S. Chen, "Multi-instance multi-label active learning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. ijcai.org, 2017, pp. 1886–1892.

[24] M. J. Huiskes and M. S. Lew, "The mir flickr retrieval evaluation," in *Proceedings of the 1st ACM international conference on Multimedia Information Retrieval*. ACM, 2008, pp. 39–43.

[25] E. Hüllermeier and J. Beringer, "Learning from ambiguously labeled examples," *Intelligent Data Analysis*, vol. 10, no. 5, pp. 419–439, 2006.

[26] T. Ishida, G. Niu, W. Hu, and M. Sugiyama, "Learning from complementary labels," in *Advances in Neural Information Processing Systems*, 2017, pp. 5639–5649.

[27] R. Jin and Z. Ghahramani, "Learning with multiple labels," in *Advances in Neural Information Processing Systems*, 2003, pp. 921–928.

[28] B. Johan, C. Bosco, and A. Mazzei, "Converting a dependency treebank to a categorial grammar treebank for italian," in *Proceedings of the 8th International Workshop on Treebanks and Linguistic Theories*. Educatt, 2009, pp. 27–38.

[29] Y. Li, L. Guo, and Z. Zhou, "Towards safe weakly supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 334–346, 2019.

[30] Y.-F. Li and D.-M. Liang, "Safe semi-supervised learning: a brief introduction," *Frontiers of Computer Science*, vol. 13, no. 4, pp. 669–676, 2019.

[31] C. H. Lin, Mausam, and D. S. Weld, "Re-active learning: Active learning with relabeling," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1845–1852.

[32] L. Liu and T. G. Dietterich, "A conditional multinomial mixture model for superset label learning," in *Advances in Neural Information Processing Systems*, 2012, pp. 548–556.

[33] W. Liu, X. Shen, H. Wang, and I. W. Tsang, "The emerging trends of multi-label learning," *CoRR*, vol. abs/2011.11197, 2020.

[34] W. Liu and I. W. Tsang, "Large margin metric learning for multi-label prediction," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 2800–2806.

[35] J. Luo and F. Orabona, "Learning from candidate labeling sets," in *Advances in Neural Information Processing Systems*, 2010, pp. 1504–1512.

[36] G. Lyu, S. Feng, T. Wang, C. Lang, and Y. Li, "GM-PLL: graph matching based partial label learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 2, pp. 521–535, 2021.

[37] Y. Ma, C. Cui, X. Nie, G. Yang, K. Shaheed, and Y. Yin, "Pre-course student performance prediction with multi-instance multi-label learning," *Science China Information Sciences*, vol. 62, no. 2, p. 29101, 2019.

[38] N. Nguyen and R. Caruana, "Classification with partial labels," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 551–559.

[39] G. Panis, A. Lanitis, N. Tsapatsoulis, and T. F. Cootes, "Overview of research on facial ageing using the fg-net ageing database," *IET Biometrics*, vol. 5, no. 2, pp. 37–46, 2016.

[40] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proceedings of the 15th IEEE International Conference on Computer Vision*, 2015, pp. 1742–1750.

[41] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 4331–4340.

[42] B. Schölkopf, "The kernel trick for distances," in *Advances in Neural Information Processing Systems*, 2000, pp. 301–307.

[43] J. Song, H. Wang, Y. Gao, and B. An, "Active learning with confidence-based answers for crowdsourcing labeling tasks," *Knowledge Based Systems*, vol. 159, pp. 244–258, 2018.

[44] L. Sun, S. Feng, T. Wang, C. Lang, and Y. Jin, "Partial multi-label learning by low-rank and sparse decomposition," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5016–5023.

[45] C. Tang and M. Zhang, "Confidence-rated discriminative partial label learning," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 2611–2617.

[46] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*. Springer, 2009, pp. 667–685.

[47] D.-B. Wang, L. Li, and M.-L. Zhang, "Adaptive graph guided disambiguation for partial label learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2019, pp. 83–91.

[48] H. Wang, W. Liu, Y. Zhao, T. Hu, K. Chen, and G. Chen, "Learning from multi-dimensional partial labels," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020, pp. 2943–2949.

[49] H. Wang, W. Liu, Y. Zhao, C. Zhang, T. Hu, and G. Chen, "Discriminative and correlative partial multi-label learning," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019, pp. 3691–3697.

[50] H. Wang, Y. Qiang, C. Chen, W. Liu, T. Hu, Z. Li, and G. Chen, "Online partial label learning," in *European Conference on Machine Learning and Knowledge Discovery in Databases*, vol. 12458, 2020, pp. 455–470.

[51] J. Wang and M.-L. Zhang, "Towards mitigating the class-imbalance problem for partial label learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2018, pp. 2427–2436.

[52] L. Wang, Z. Ding, and Y. Fu, "Adaptive graph guided embedding for multi-label annotation." in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 2798–2804.

[53] Q.-W. Wang, Y.-F. Li, and Z.-H. Zhou, "Partial label learning with unlabeled data," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 3755–3761.

[54] J.-H. Wu and M.-L. Zhang, "Disambiguation enabled linear discriminant analysis for partial label dimensionality reduction," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2019, pp. 416–424.

[55] X. Wu and M. Zhang, "Towards enabling binary decomposition for partial label learning," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, pp. 2868–2874.

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

[57] N. Xu, J. Lv, and X. Geng, "Partial label learning via label enhancement," in *The Thirty-Third AAAI Conference on Artificial Intelligence*, 2019, pp. 5557–5564.

[58] S. Yan, K. Chaudhuri, and T. Javidi, "Active learning from noisy and abstention feedback," in *Proceedings of the 53rd Annual Allerton Conference on Communication, Control, and Computing*, 2015, pp. 1352–1357.

[59] S. Yan, K. Chaudhuri, and T. Javidi, "Active learning from imperfect labelers," in *Advances in Neural Information Processing Systems*, 2016, pp. 2128–2136.

[60] S. Yan, K. Chaudhuri, and T. Javidi, "Active learning with logged data," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, 2018, pp. 5517–5526.

[61] J. Yang, T. Drake, A. C. Damianou, and Y. Maarek.

[62] Y. Ye and E. Tse, "An extension of karmarkar's projective algorithm for convex quadratic programming," *Mathematical Programming*, vol. 44, no. 1-3, pp. 157–179, 1989.

[63] F. Yu and M.-L. Zhang, "Maximum margin partial label learning." *Machine Learning*, vol. 106, no. 4, pp. 573–593, 2017.

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

[65] Z. Zeng, S. Xiao, K. Jia, T.-H. Chan, S. Gao, D. Xu, and Y. Ma, "Learning by associating ambiguously labeled images," in *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 708–715.

[66] L. Zhang, Q. Zhang, B. Du, J. You, and D. Tao, "Adaptive manifold regularized matrix factorization for data clustering," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 3399–3405.

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

[68] M.-L. Zhang, F. Yu, and C.-Z. Tang, "Disambiguation-free partial label learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 10, pp. 2155–2167, 2017.

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

[70] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[71] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in Neural Information Processing Systems*, 2015, pp. 649–657.

[72] D. Zhou, Z. Zhang, M.-L. Zhang, and Y. He, "Weakly supervised pos tagging without disambiguation," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 17, no. 4, p. 35, 2018.

[73] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

[74] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, "Multi-instance multi-label learning," *Artificial Intelligence*, vol. 176, no. 1, pp. 2291–2320, 2012.

[75] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–130, 2009.