Adaptive Graph Guided Disambiguation for Partial Label Learning

Deng-Bao Wang[†], Li Li[†], Min-Ling Zhang[‡♮*]

[†]College of Computer and Information Science, Southwest University, Chongqing, China
[‡]School of Computer Science and Engineering, Southeast University, Nanjing, China
[♮]Key Lab. of Computer Network and Information Integration (Southeast University), MOE, China
dengbaowang@gmail.com, lily@swu.edu.cn, zhangml@seu.edu.cn

ABSTRACT

Partial label learning 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. The common strategy to train predictive model is *disambiguation*, i.e. differentiating the modeling outputs of individual candidate labels so as to recover ground-truth labeling information. Recently, feature-aware disambiguation was proposed to generate different labeling confidences over candidate label set by utilizing the graph structure of feature space. However, the existence of noise and outliers in training data makes the similarity derived from original features less reliable. To this end, we proposed a novel approach for partial label learning based on *adaptive graph guided disambiguation* (PL-AGGD). Compared with fixed graph, adaptive graph could be more robust and accurate to reveal the intrinsic manifold structure within the data. Moreover, instead of the two-stage strategy in previous algorithms, our approach performs label disambiguation while training the predictive model simultaneously. Specifically, we present a unified framework that makes ground-truth labeling confidences, similarity graph and model parameters can be jointly optimized to achieve the best results. Extensive experiments show that PL-AGGD performs favorably against state-of-the-art partial label learning approaches.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Information systems** → *Data mining*.

KEYWORDS

Partial label learning, adaptive graph, disambiguation

ACM Reference Format:
Deng-Bao Wang, Li Li, and Min-Ling Zhang. 2019. Adaptive Graph Guided Disambiguation for Partial Label Learning. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330840

*Min-Ling Zhang is the corresponding author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330840

1 INTRODUCTION

Multi-class classification is an important problem in various real-world applications, such as natural language processing, computer vision, natural language processing, web advertising, etc [8, 31]. In ordinary multi-class classification problems, each training example is associated with a label which specifies the class the example belongs to. Although learning from this kind of supervised datasets is effective, collecting labeled data is costly and thus a critical bottleneck in real-world classification tasks. For this reason, learning from less expensive data has been extensively studied in the last decades [7, 14, 21, 33].

Partial label learning [4, 11, 18, 23, 29] 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 is motivated in domains in which a large number of ambiguously labeled examples are available while it is expensive to acquire explicit labels, and arises in many real-world tasks such as automatic image annotation [6, 26], web mining [17], ecoinformatics [16], etc.

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 l possible class labels. Suppose we have the partial label training set $\mathcal{D} = \{(\boldsymbol{x}_i, S_i) | 1 \leq i \leq m\}$ where $\boldsymbol{x}_i \in \mathcal{X}$ is a d dimensional feature vector and S_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 setting is that the ground truth label y_i of $\boldsymbol{x}_i$ resides in S_i but it is not accessible to algorithm during training phase.

The common strategy to train predictive model is disambiguation, i.e. differentiating the modeling outputs of individual candidate labels so as to recover ground-truth labeling information. There are mainly two learning frameworks, including the averaging-based framework and the identification-based framework. For averaging-based disambiguation, all the potential labels of each example are treated equally and the prediction is made by averaging their modeling outputs [7, 13, 28]. For identification-based disambiguation, the ground-truth label is regarded as latent variable and identified through iterative refining procedure [5, 15, 16, 18].

Recently, feature-aware disambiguation method [29] was proposed to generate different labeling confidences over candidate label set by utilizing the graph structure of feature space. To facilitating the disambiguation process, feature-aware disambiguation makes use of the smoothness assumption that examples close to each other in the feature space will tend to share identical label in the label space. The approach proposed in [29] works in a two-stage

framework which firstly generates latent labeling confidences over candidate label set and then learns a multi-class model by fitting a multi-output regressor with the generated labeling confidences.

Although the results in [29] have shown the effectiveness of feature-aware disambiguation for partial label learning, one potential drawback of this strategy lies in that real world data always contain lots of noise and outliers that make the similarity graph obtained by original features cannot be fully relied. To this end, a novel approach for partial label learning is proposed in this paper based on *adaptive graph guided disambiguation* (PL-AGGD). Compared with fixed graph, adaptive graph could be more robust and accurate to reveal the intrinsic manifold structure within the data. Moreover, instead of the two-stage strategy in previous learning algorithms, our approach performs label disambiguation while training the predictive model simultaneously. Specifically, we present a unified framework that makes ground-truth labeling confidences, similarity graph and model parameters can be jointly optimized to achieve the best results, and we adopt an alternating method to solve this optimization problem. Extensive experiments on controlled UCI data sets as well as real-world PL data sets show that PL-AGGD performs favorably against other partial label learning approaches.

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

In partial label learning, the ground truth label of an example resides in a candidate label set but it is not accessible to algorithm during training phase, therefore it can be regarded as a weakly-supervised learning framework. To put the partial label learning problem into perspective, it is useful to lay out several related learning scenarios, including semi-supervised learning, multi-instance learning, multi-label learning and multi-instance multi-label learning. In semi-supervised learning [33], training examples are either explicitly labeled or unlabeled, while in partial label learning training examples are partially labeled. In multiple-instance learning [1, 2], examples are not individually labeled but grouped into bags, while in partial label learning labels are assigned at the level of individual instances. In multi-label learning [10, 22, 30], each example is assigned 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 [32] can be considered as a generalized version of multi-instance learning and multi-label learning, where training instances are associated with not only multiple instances but also multiple labels. Partial label learning ignores any bag structure and views each instance independently, while 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 [16].

Existing partial label learning approaches try to solve the learning task by disambiguating the candidate label set, which can be achieved in two basic strategies. One intuitive strategy is to treat all the candidate labels of an example in an equal manner and the prediction is made by averaging their modeling outputs. For predictive models $g(\boldsymbol{x}, \mathbf{W})$ ($\mathbf{W}$ denotes the model parameters), the averaged output over all candidate labels, i.e. $\frac{1}{|S_i|}\sum_{y\in S_i} g_y(\boldsymbol{x}_i, \mathbf{W})$, is distinguished from the outputs from non-candidate labels, i.e. $g_y(\boldsymbol{x}_i, \mathbf{W})$ ($y \notin S_i$) [7]. For non-parametric models, the predicted label for test example is determined by voting among the candidate labels of its neighboring examples [13, 28]. One potential drawback of the averaging-based disambiguation strategy lies in that the essential modeling output from ground-truth label might be overwhelmed by the distractive outputs from false positive labels.

Though the averaging strategy is intuitive and easy to be implemented, its effectiveness is largely affected by the false positive labels whose outputs would overwhelm the essential output yielded by the ground-truth label. Therefore, another strategy is to disambiguate the candidate label set by identifying the ground-truth label. Existing approaches following this strategy regard the ground-truth label as latent variable which is identified as: $\hat{y}_i = \arg\max_{y\in S_i} g_y(\boldsymbol{x}_i, \mathbf{W})$. Generally, the latent variable and model parameters are refined iteratively via EM procedure which optimizes objective function defined according to the maximum likelihood criterion: $\sum_{i=1}^{m} \log(\sum_{y\in S_i} g_y(\boldsymbol{x}_i, \mathbf{W}))$ [15, 16], or the maximum margin criterion: $\sum_{i=1}(\max_{y\in S_i} g_y(\boldsymbol{x}_i, \mathbf{W}) - \max_{y\notin S_i} g_y(\boldsymbol{x}_i, \mathbf{W}))$ [18, 25]. One potential 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.

Recently proposed feature-aware disambiguation [9, 29] makes use of local manifold information from the feature space. The candidate labels are disambiguated in the form of normalized labeling confidences, which differs from both averaging-based disambiguation and identification-based disambiguation. One potential drawback of this strategy lies in that real world data always contain lots of noise and outliers that make the similarity graph obtained by original features less reliable [19, 24, 27]. In the next section, we will introduce a novel approach which disambiguates the candidate labels with an adaptive graph. By adopting an alternating optimization algorithm, our approach can iteratively refine the labeling confidences, similarity graph and model parameters.

3 THE PROPOSED APPROACH

In this section, we introduce the proposed approach PL-AGGD. We first present an unified framework which aims to perform disambiguation, adaptive graph optimization and model learning simultaneously. After that, an efficient optimization procedure is introduced to tackle the proposed framework.

3.1 Adaptive Graph Guided Disambiguation

We 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 $s_{ij} = 0$ if $(\boldsymbol{x}_i, \boldsymbol{x}_j) \notin \mathcal{E}$.

For each training example $(\boldsymbol{x}_i, S_i)$, we aim to generate a normalized real-valued vector $\boldsymbol{f}_i \in \mathbb{R}^q$ 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) $\sum_{y_{il}=1} f_{il}=1$, (ii) $f_{il}\geq 0$ $(\forall y_{il}=1)$ and (iii) $f_{il}=0$ $(\forall y_{il}=0)$. Here the second constraint implies that the latent ground-truth label resides in the candidate label set, and the third constraint guarantees that the labeling confidence of each non-candidate label must be 0. Once the normalized labeling confidence vectors have been generated, the original partial label training set $\mathcal{D}$ has been transformed into its disambiguated counterpart: $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, \boldsymbol{f}_i) | 1 \leq i \leq m\}$. Then the predictive model can be induced by performing multi-output regression based on the disambiguation results $\widetilde{\mathcal{D}}$.

Feature-aware disambiguation differentiates the candidate label set by exploiting graph structure of the feature space. Specifically, the similarity graph weight matrix $\mathbf{S}$ is constructed by solving the following linear least square problem:

$$\min_{\mathbf{S}} \sum_{j=1}^{m} \left\| \boldsymbol{x}_j - \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{x}_i \right\|_2^2 \tag{1}$$

$$\text{s.t. } \mathbf{S}\mathbf{1}_m = \mathbf{1}_m,\ s_{ij} \geq 0\ (\forall(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}),\ s_{ij} = 0\ (\forall(\boldsymbol{x}_i, \boldsymbol{x}_j) \notin \mathcal{E})$$

where $\mathbf{1}_m$ is an all 1 vector with size m.

Then by exploiting the smoothness assumption that the manifold structure in the feature space should also be preserved in the label space, the labeling confidence matrix $\mathbf{F} = [\boldsymbol{f}_1, \boldsymbol{f}_2, ..., \boldsymbol{f}_m]^\top \in \mathbb{R}^{m\times q}$ can be determined by solving the following problem:

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

In previous feature-aware disambiguation, the problem (1) and (2) are solved independently, thus the disambiguation process is actually guided by a fixed graph. It is challenging for the fixed similarity graph to reveal the intrinsic structure within the data since noise and outliers widely exist, thus we consider to get the similarity weight matrix in an adaptive way. We obtain the similarity weight matrix and simultaneously differentiate the labeling confidence to achieve better disambiguation results. By solving the problem (1) and (2) jointly, the similarity graph weights are not only determined by the distance information of features, but also adjusted based on the feedback of label space. Compared with fixed graph, adaptive graph could be more robust and accurate.

Furthermore, instead of the two-stage (disambiguation and model induction) strategy in previous algorithms, we perform label disambiguation while training the predictive model simultaneously. Thus the labeling confidence can be optimized by considering both the manifold assumption and model outputs. We present a unified framework to make labeling confidence, similarity weights and predictive model jointly optimized to achieve the best results. 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:

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

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

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

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. $\mathbf{0}_{m\times m}$ is the $m \times m$ all 0 matrix. λ is the regularization parameter, and μ, γ are the trade-off parameters for reconstruction loss of label and feature space.

Table 1: The pseudo-code of PL-AGGD

Input:

$\mathcal{D}$: the partially labeled training set $\{(\boldsymbol{x}_i, S_i) | 1 \leq i \leq m\}$

k: the number of nearest neighbors

μ, γ, λ: the trade-off parameters

$\boldsymbol{x}$: the unseen instance

Output:

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

Process:

Calculate the kernel matrix $\mathbf{K}=[\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{m\times m}$ $(1 \leq i, j \leq m)$;
Set the similarity graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$;
Initial $\mathbf{S}$ by solving the problem (1);
Initial $\mathbf{F}$ by solving the problem (2);
repeat
  Update $\mathbf{A}$ according to (17);
  Update model outputs $\mathbf{H} = \frac{1}{2\lambda}\mathbf{K}\mathbf{A}$;
  for $j = 1$ **to** m **do**
    Update the j-th column of $\mathbf{S}$ by solving (7);
  end for
  Update $\mathbf{F}$ by solving (9) or iteratively solving a series subproblems shown as (11);
until convergence or the maximum number of iterations
return y according to (18).

3.2 Alternative Optimization

Since problem (4) contains three sets of variables with different regularizations and constraints, it is hard to be tackled directly. Thus we apply the alternative optimization approach to solve this problem. Specifically, we iteratively optimize each set of variables by fixing other sets of variables until convergence or the maximum number of iterations is reached.

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

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

The similarity vector of each sample is independent, thus we 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:

$$\min_{\boldsymbol{s}_j} \mu\left\|\boldsymbol{f}_j - \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{f}_i\right\|_2^2 + \gamma\left\|\boldsymbol{x}_j - \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{E}} s_{ij}\boldsymbol{x}_i\right\|_2^2 \quad \text{s.t. } \boldsymbol{s}_j^\top \mathbf{1}_m = 1,\ \mathbf{0}_m \le \boldsymbol{s}_j \le \boldsymbol{n}_j \tag{6}$$

Note that for $\boldsymbol{s}_j$, there are only k non-negative elements 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 $\mathbf{D}^{fj} = [\boldsymbol{f}_j - \boldsymbol{f}_{N_j(1)}, \boldsymbol{f}_j - \boldsymbol{f}_{N_j(2)}, ..., \boldsymbol{f}_j - \boldsymbol{f}_{N_j(k)}]^\top \in \mathbb{R}^{k\times q}$ and matrix $\mathbf{D}^{xj} = [\boldsymbol{x}_j - \boldsymbol{x}_{N_j(1)}, \boldsymbol{x}_j - \boldsymbol{x}_{N_j(2)}, ..., \boldsymbol{x}_j - \boldsymbol{x}_{N_j(k)}]^\top \in \mathbb{R}^{k\times d}$. Then the optimization problem (6) can be re-written as follows:

$$\min_{\hat{\boldsymbol{s}}_j} \hat{\boldsymbol{s}}_j^\top(\mu\mathbf{G}^{fj} + \gamma\mathbf{G}^{xj})\hat{\boldsymbol{s}}_j \quad \text{s.t. } \hat{\boldsymbol{s}}_j^\top \mathbf{1}_k = 1,\ \mathbf{0}_k \le \hat{\boldsymbol{s}}_j \le \mathbf{1}_k \tag{7}$$

where $\mathbf{G}^{fj} = \mathbf{D}^{fj}(\mathbf{D}^{fj})^\top \in \mathbb{R}^{k\times k}$ and $\mathbf{G}^{xj} = \mathbf{D}^{xj}(\mathbf{D}^{xj})^\top \in \mathbb{R}^{k\times k}$ are two Gram matrices respectively corresponding to label space and feature space. 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:

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

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.

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 $\tilde{\boldsymbol{y}} = vec(\mathrm{Y}) \in \{0,1\}^{mq}$. We define a square matrix $\mathbf{T} = \mathbf{S}\mathbf{S}^\top + (\mathbf{S}^\top\mathbf{1}_{m\times m}\mathbf{S}) \odot \mathbf{I}_{m\times m} - 2\mathbf{W} + \frac{1}{\mu}\mathbf{I}_{m\times m}$ where $\odot$ is the Hadamard product (the entrywise product) and $\mathbf{I}_{m\times m}$ is an identity matrix with m rows and m columns. Then the optimization problem (8) can be written as follows:

$$\min_{\tilde{\boldsymbol{f}}} \frac{1}{2}\tilde{\boldsymbol{f}}^\top(\boldsymbol{\Lambda}+\boldsymbol{\Lambda}^\top)\tilde{\boldsymbol{f}} - \frac{2}{\mu}\tilde{\boldsymbol{h}}^\top\tilde{\boldsymbol{f}} \quad \text{s.t. } \mathbf{0}_{mq} \le \tilde{\boldsymbol{f}} \le \tilde{\boldsymbol{y}},\ \sum_{j=1,j\%m=i}^{mq}\tilde{\boldsymbol{f}}_j = 1\ (\forall\, 0\le i\le m-1) \tag{9}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{mq\times mq}$ is defined as follows:

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

The optimization problem (9) also corresponds a standard Quadratic Programming problem, which can be solved by off-the-shelf QP tools. The second constraint of problem (9) 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 [29] we employ alternating optimization strategy for large scale datasets. For labeling confidence vector $\boldsymbol{f}_j$, it can be optimized by fixing all other labeling confidence vectors:

$$\min_{\boldsymbol{f}_j} (t_{jj} + \frac{1}{\mu})\boldsymbol{f}_j^\top\boldsymbol{f}_j + \left(\sum_{i=1,i\neq j}^{m}(t_{ij}+t_{ji})\boldsymbol{f}_i^\top - \frac{2}{\mu}\boldsymbol{h}_i^\top\right)\boldsymbol{f}_j \quad \text{s.t. } \mathbf{0}_q \le \boldsymbol{f}_j \le \boldsymbol{y}_j,\ \boldsymbol{f}_j\mathbf{1}_q = 1 \tag{11}$$

Now the problem (9) is transformed into a series of QP subproblems with q variables and $q+1$ constraints which can be solved more efficiently.

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

$$\min_{\mathbf{W}} \sum_i \left\|\mathbf{W}^\top\boldsymbol{x}_i - \boldsymbol{f}_i\right\|_2^2 + \lambda\|\mathbf{W}\|_{\mathrm{F}}^2 \tag{12}$$

This regularized least squares problem is simple and can be easily solved by gradient descent method. To achieve better performance of the predictive model, we can further facilitate a kernel extension for the general nonlinear case. Let $\phi(\bullet) : \mathbb{R}^d \to \mathbb{R}^h$ denotes the feature mapping that maps the feature space to some higher dimensional Hilbert space with h dimensions. The predictive function with kernel extension can be represented as $g(\boldsymbol{x}_i, \mathbf{W}) = \mathbf{W}^\top\phi(\boldsymbol{x}_i)$. Then we have the following problem:

$$\min_{\mathbf{W}} \sum_i \|\boldsymbol{\epsilon}_i\|_2^2 + \lambda\|\mathbf{W}\|_{\mathrm{F}}^2 \quad \text{s.t. } \boldsymbol{f}_i = \mathbf{W}^\top\phi(\boldsymbol{x}_i) - \boldsymbol{\epsilon}_i \tag{13}$$

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 as the following matrix form:

$$\min_{\mathbf{W}} tr(\mathbf{E}^\top\mathbf{E}) + \lambda tr(\mathbf{W}^\top\mathbf{W}) \quad \text{s.t. } \mathbf{F} = \boldsymbol{\Phi}\mathbf{W} - \mathbf{E} \tag{14}$$

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

$$\mathcal{L}(\mathbf{W},\mathbf{E},\mathbf{A}) = tr(\mathbf{E}^\top\mathbf{E}) + \lambda tr(\mathbf{W}^\top\mathbf{W}) - tr(\mathbf{A}^\top(\boldsymbol{\Phi}\mathbf{W}-\mathbf{E}-\mathbf{F})) \tag{15}$$

where $\mathbf{A} \in \mathbb{R}^{m\times q}$ stores the Lagrange multipliers. We know that the optimum solution of (15) need to satisfy the following conditions:

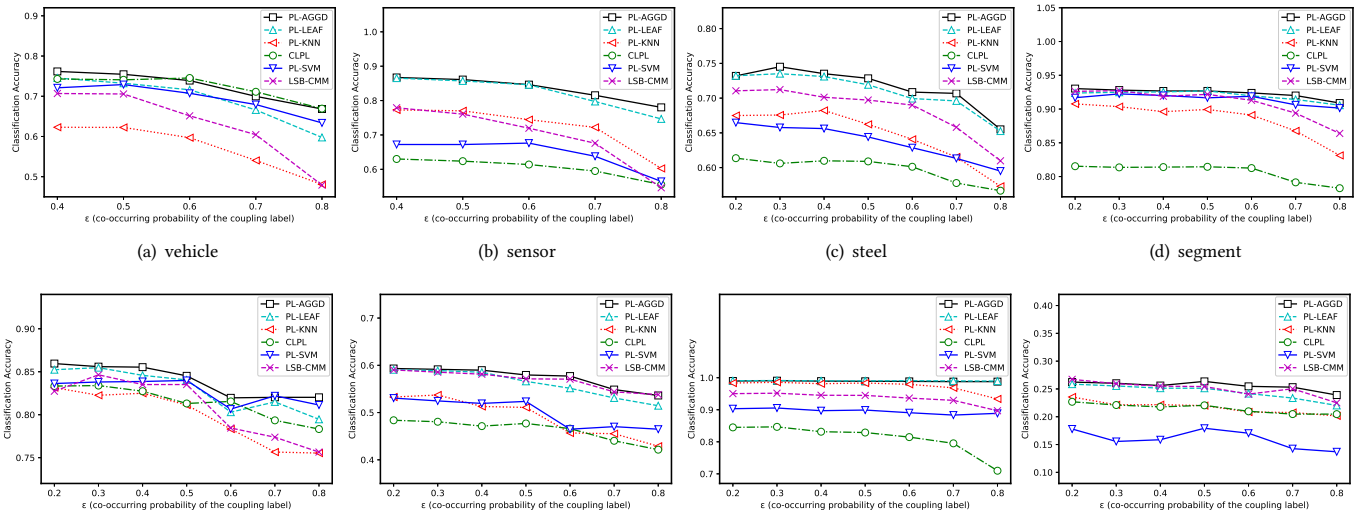
$$\frac{\partial\mathcal{L}}{\partial\mathbf{W}} = 2\lambda\mathbf{W} - \boldsymbol{\Phi}^\top\mathbf{A} = \mathbf{0} \quad \frac{\partial\mathcal{L}}{\partial\mathbf{E}} = 2\mathbf{E} + \mathbf{A} = \mathbf{0} \quad \frac{\partial\mathcal{L}}{\partial\boldsymbol{\Psi}} = \boldsymbol{\Phi}\mathbf{W} - \mathbf{E} - \mathbf{F} = \mathbf{0} \tag{16}$$

From these linear equations of we can obtain:

$$\mathbf{W} = \frac{\boldsymbol{\Phi}^\top\mathbf{A}}{2\lambda} \qquad \mathbf{A} = \left(\frac{1}{2\lambda}\mathbf{K} + \frac{1}{2}\mathbf{I}_{m\times m}\right)^{-1}\mathbf{F} \tag{17}$$

where $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ with its element $k_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where $\kappa(\cdot,\cdot)$ is the kernel function. 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 such kernel function, the modeling output matrix is denoted by $\mathbf{H} = \boldsymbol{\Phi}\mathbf{W} = \frac{1}{2\lambda}\mathbf{K}\mathbf{A}$. And the predicted label of the test example $\boldsymbol{x}$ by our approach is given as:

$$y^* = \arg\min_k \sum_{i=1}^{m} a_{ik}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) \tag{18}$$



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

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

Figure 1: Classification accuracy of each comparing algorithm with the varying ϵ (co-occurring probability of the coupling label) and one false positive candidate label ($r = 1$).

The pseudo code of PL-AGGD is summarized in Table 1. We can observe that the first process of our approach is to initialize **S** and **F** (Steps 2-4) by adopting feature-aware disambiguation. After that, the alternative optimization strategy is adopted to learn an adaptive graph to guide the label disambiguation (Steps 5-12) by learning the predictive model simultaneously. Finally, the unseen instance is classified based on the learned model.

4 EXPERIMENTS

4.1 Experimental Setup

In this section, we conduct two series of comparative experiments based on controlled UCI data sets as well as real-world partial label data sets. Characteristics of the controlled UCI data sets and real-world data sets are summarized in Table 2 and Table 5 respectively.

Following the widely-used controlling protocol in previous partial label learning research [5, 7, 16], an artificial PL data set can be generated from a multi-class UCI data set with controlling parameters p, r and ϵ. Here, p controls the proportion of examples which are partially labeled (we simply set $p = 1$ across all our experiments). r controls the number of false positive labels in the candidate label set (i.e. $|S_i| = r + 1$), and in our setting the ambiguous labels in the training set are generated without replacement. ϵ is the ambiguity degree which controls the co-occurring probability between one coupling candidate label and the ground-truth label. This is achieved by first 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). In addition to artificial data sets, a number of real-world partially labeled data sets have been collected from several task domains (see in Table 5) including `Lost` [6], `Soccer Player` [26], `Yahoo! News` [12] from automatic face naming, `MSRCv2` [16] from object classification, `FG-NET` [20] from facial age estimation, and `BirdSong` [3] from bird song classification. For the task of automatic face naming, faces cropped from an image or video are represented as instances while names extracted from the associated captions or subtitles are regarded as candidate labels. Specifically, by retaining top frequent names from the Labeled `Yahoo! News` data set. For the task of object classification, image segmentations are represented as instances while objects appearing within the same image are regarded as candidate labels. For the task of facial age estimation, human faces with landmarks are represented as instances while ages annotated by ten crowdsourced labelers together with the ground-truth age are regarded as candidate labels. For the task of 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. The average number of candidate labels (avg. #PLs) for each real-world PL data set is also recorded in Table 5.

Table 2: 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

Five state-of-the-art partial label learning algorithms are employed for comparative studies, each implemented with hyper parameters setup suggested in respective literatures:

Table 3: Classification accuracy (mean±std) of each comparing algorithm on the controlled UCI data sets with the 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).

	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•
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•
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•
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•

- PL-KNN [13]: A k-nearest neighbor approach to partial label learning which conducts averaging-based disambiguation [suggested setup: $k = 10$];
- CLPL [7]: An averaging-based disambiguation approach based on a convex learning formulation which minimizes a appropriate loss function for the partial label setting [suggested setup: SVM with squared hinge loss];
- PL-SVM [18]: A maximum margin approach to partial label learning which conducts identification-based disambiguation [suggested setup: regularization parameter pool with $\{0.01, 0.1, 1, 10, 100\}$];
- LSB-CMM [16]: A maximum likelihood approach to partial label learning which conducts identification-based disambiguation [suggested setup: mixture components = q].
- PL-LEAF [29]: Two-stage approach which learns from partial label examples based on feature-aware disambiguation [$k = 10, C_1 = 10$ and $C_2 = 1$].

For PL-AGGD, the parameters employed are set as $k = 10$, $T = 20$, $\lambda = 1$, $\mu = 1$ and $\gamma = 0.05$. The sensitivity analysis of parameter configuration is conducted in Subsection 4.3. We perform ten runs of 50%/50% random train/test splits on each artificial as well as real-world PL data set, and the mean accuracies (with standard deviations) are recorded for all algorithms.

4.2 Experimental Results

Controlled Data Sets Figure 1 illustrates the classification accuracy of each comparing algorithm on the controlled UCI data sets as the co-occurring probability ϵ varies 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).

Table 4: Win/tie/loss counts (pairwise t-test at 0.05 significance level) on the classification performance of PL-AGGD against each comparing algorithm.

	PL-AGGD **against**				
	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
[Figure 1]	52/0/0	47/5/0	45/7/0	38/13/1	29/20/3
[Table 3]	23/0/0	23/0/0	10/2/1	16/4/3	13/8/2
In total	**75/0/0**	**70/5/0**	**65/9/1**	**54/17/4**	**42/28/5**

Table 5: Characteristics of the real world data sets.[1]

Data set	#examples	#features	#classes	avg. #PLs	Task Domain
Lost	1122	108	16	2.23	*automatic face naming* [6]
MSRCv2	1758	48	23	3.16	*object classification* [16]
FG-NET	1002	262	78	7.48	*facial age estimation* [20]
Birdsong	4998	38	13	2.18	*bird song classification* [3]
Soccer Player	17472	279	171	2.09	*automatic face naming* [26]
Yahoo! News	22991	163	219	1.91	*automatic face naming* [12]

Table 6: 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).

	PL-AGGD	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
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
FG-NET	0.076±0.005	0.038±0.009•	0.053±0.012•	0.060±0.013•	0.053±0.007•	0.067±0.011•
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•
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•
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•
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•

In Table 3, we vary the ambiguity size: the number of extra labels associated with each example. 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 Figure 1 and Table 3, PL-AGGD achieves superior or competitive performance against the comparing algorithms. Out of the 75 statistical tests on all 8 UCI data sets (52 configurations in Figure 1 and 23 configurations in Table 3), it is shown that:

- Comparing to averaging-based disambiguation approaches, PL-AGGD achieves superior performance against PL-KNN and CLPL in in most cases.
- Comparing to identification-based disambiguation approaches, PL-AGGD achieves superior performance against PL-SVM and LSB-CMM in 86.7% and 72% cases respectively. Furthermore, the performance of PL-AGGD is inferior to LSB-CMM in only 1 cases and has been outperformed by PL-SVM in only 4 cases.
- Comparing to existing feature-aware approach PL-LEAF, PL-AGGD achieves superior performance in 56% and outperformed by PL-LEAF in only 5 cases. Furthermore, from Figure 1 and Table 3 we can observe that PL-AGGD is more robust when class ambiguity degree ϵ or ambiguity size r is large.

Real-World Data Sets Table 6 reports the classification accuracy of each comparing algorithm on the real-world partial label data sets. Note that the average number of candidate labels (avg. #PLs) of the dataset FG-NET is quite large, which causes an extremely low classification accuracy of each algorithm based on training examples with partial labels. The common evaluation criteria on this task is mean average error (MAE) between the predicted age and the ground-truth age. In Table 6, for better evaluation of this facial age estimation task, two extra experiments are conducted on the dataset FG-NET where a test example is considered to be correctly classified if the difference between the predicted age and the ground-truth age is no more than 3 years (MAE3) or 5 years (MAE5). As shown in Table 6, it is obvious that PL-AGGD significantly outperforms all the counterpart algorithms on these real-world datasets expect for PL-SVM on Yahoo! News and PL-LEAF on MSRCv2. Furthermore, our approach is never outperformed by any comparing algorithm.

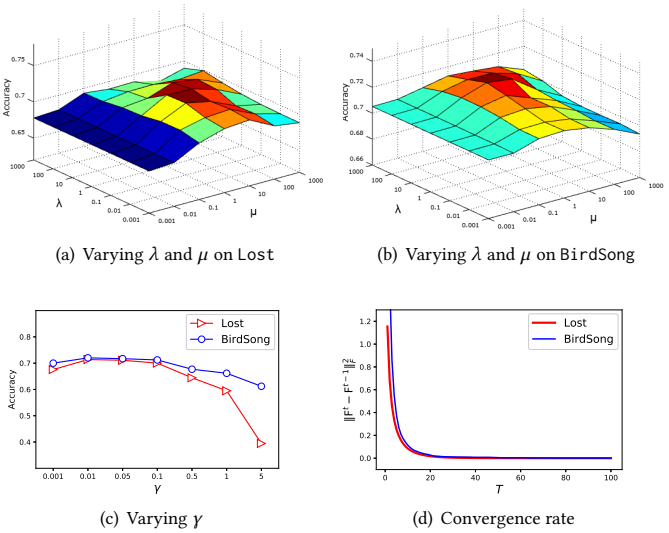
4.3 Further Analysis

Transductive Accuracy In addition to Table 6 reporting inductive performance on test examples, it is also interesting to study the transductive performance of each comparing algorithm on classifying training examples [7, 29]. Transductive performance of the partial label learning algorithm reflects its ability in disambiguating the candidate label set. For PL-AGGD and PL-LEAF, the generated labeling confidence vector $\boldsymbol{f}_i$ can be used to predict the ground-truth label of training example x_i as $\hat{y}_i = \arg\max_{y_k \in S_i} \boldsymbol{f}_{ik}$. For other approach, the ground-truth label is predicted by consulting the candidate label set S_i, i.e. predicting $\hat{y}_i \in S_i$ with largest modeling output. Accordingly, Table 7 reports the transductive accuracy of each comparing algorithm together with the outcomes of pairwise t-tests at 0.05 significance level. Out of the 40 statistical tests (8 data sets × 5 comparing algorithm), it is shown that:

[1]These data sets are publicly-available at: http://cse.seu.edu.cn/PersonalPage/zhangml/Resources.htm#partial_data

Table 7: 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).

	PL-AGGD	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF
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
FG-NET	0.142±0.007	0.171±0.021○	0.160±0.010○	0.155±0.010	0.128±0.010•	0.133±0.008•
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•
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•
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•
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 and μ on Lost

(b) Varying λ and μ on BirdSong

(c) Varying γ

(d) Convergence rate

Figure 2: 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 Lost and BirdSong by varying γ; (d) Convergence curves of the labeling scores on Lost and BirdSong.

- Comparing to averaging-based disambiguation approaches, PL-AGGD is outperformed by PL-KNN on the FG-NET data set, and outperformed by CLPL on the Lost and FG-NET data set. In 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 sets. In the rest statistical tests, the performance of PL-AGGD is superior or at least comparable to PL-SVM and LSB-CMM.
- Comparing to existing feature-aware approach PL-LEAF, the disambiguation performance of PL-AGGD is superior or at least comparable on all data sets.

Parameter Sensitivity We also study the sensitivity of PL-AGGD with respect to its three parameters λ, μ and γ. Figure 2 shows the performance of PL-AGGD under different parameter configurations on Lost and BirdSong. As shown in Figure 2(a) and 2(b), when μ is small (it means that the manifold information of label space is hardly exploited when update $\mathbf{F}$), PL-AGGD gives a poor performance. As μ increases, PL-AGGD start to take into consideration the manifold information, and the classification accuracy increases. However, if μ is sufficiently large, the classification accuracy will drop dramatically. For λ, the perform would also be at a low level when it is too small or large. In practice, we suggest users select λ and μ around 1 for safety. Another trade-off parameter γ aims to control the model complexity. The classification accuracy curve of varying γ (in Figure 2(c)) obviously accords with the fact that it is important to balance between overfitting and underfitting. In practice it can be manually searched in advance of training.

Figure 2(d) illustrates the convergence of PL-AGGD by using difference between labeling confidence matrix (measured by L_2 norm $\|\mathbf{F}^{(t)} - \mathbf{F}^{(t-1)}\|_F^2$) of two adjacent iterations. We can see that the labeling confidence scores converge with increasing number of iterations ($\mathbf{F}$ becomes convergent when T reaches 20).

5 CONCLUSION

In this paper, the problem of partial label learning is studied and a novel approach based on adaptive graph guided disambiguation is proposed. Instead of employing a fixed similarity graph in previous feature-aware disambiguation strategy, our proposed approach PL-AGGD aims to learn from partial label examples by performing label disambiguation while training the predictive model simultaneously. Extensive comparative studies clearly validate that adaptive graph guided disambiguation could be more robust and accurate to reveal the intrinsic manifold structure within the data.

The proposed approach PL-AGGD perform manifold structure discovery on original feature space, it would be interesting to integrate the feature embedding technique into PL-AGGD to achieve better performance. Besides, it would also be valuable to investigate the idea of adaptive graph to other weakly supervised learning setting, such as semi-supervised learning, incomplete multi-label learning, etc.

REFERENCES

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

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in neural information processing systems*, pages 577–584, 2003.

[3] 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*, pages 534–542, 2012.

[4] C.-H. Chen, V. M. Patel, and R. Chellappa. Learning from ambiguously labeled face images. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1653–1667, 2018.

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

[6] T. Cour, B. Sapp, C. Jordan, and B. Taskar. Learning from ambiguously labeled images. In *n Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 919–926, 2009.

[7] T. Cour, B. Sapp, and B. Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12(5):1501–1536, 2011.

[8] 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 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

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

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

[11] 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*, 48(3):967–978, 2018.

[12] M. Guillaumin, J. Verbeek, and C. Schmid. Multiple instance metric learning from automatically labeled bags of faces. In *European conference on Computer Vision*, pages 634–647, 2010.

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

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

[15] R. Jin and Z. Ghahramani. Learning with multiple labels. In *Advances in neural information processing systems*, pages 921–928, 2003.

[16] L. Liu and T. G. Dietterich. A conditional multinomial mixture model for superset label learning. In *Advances in neural information processing systems*, pages 548–556, 2012.

[17] J. Luo and F. Orabona. Learning from candidate labeling sets. In *Advances in neural information processing systems*, pages 1504–1512, 2010.

[18] N. Nguyen and R. Caruana. Classification with partial labels. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–559, 2008.

[19] F. Nie, W. Zhu, X. Li, et al. Unsupervised feature selection with structured graph optimization. 2016.

[20] G. Panis, A. Lanitis, N. Tsapatsoulis, and T. F. Cootes. Overview of research on facial ageing using the fg-net ageing database. *Iet Biometrics*, 5(2):37–46, 2016.

[21] 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 IEEE international conference on computer vision*, pages 1742–1750, 2015.

[22] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer, 2009.

[23] J. Wang and M.-L. Zhang. Towards mitigating the class-imbalance problem for partial label learning. pages 2427–2436, 2018.

[24] 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*, pages 2798–2804, 2018.

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

[26] 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 IEEE Conference on Computer Vision and Pattern Recognition*, pages 708–715, 2013.

[27] 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*, pages 3399–3405, 2017.

[28] 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*, pages 4048–4054, 2015.

[29] 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*, pages 1335–1344, 2016.

[30] M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.

[31] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[32] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.

[33] X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.