Partial Label Learning via Feature-Aware Disambiguation

Min-Ling Zhang [1,2] Bin-Bin Zhou [1,2] Xu-Ying Liu [1,2]
[1] School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
{zhangml, zhoubinbin, liuxy}@seu.edu.cn

ABSTRACT

Partial label learning deals with the problem where each training example is represented by a feature vector while associated with a set of *candidate* labels, among which only one label is valid. To learn from such ambiguous labeling information, the key is to try to disambiguate the candidate label sets of partial label training examples. Existing disambiguation strategies work by either identifying the ground-truth label iteratively or treating each candidate label equally. Nonetheless, the disambiguation process is generally conducted by focusing on manipulating the label space, and thus ignores making full use of potentially useful information from the feature space. In this paper, a novel two-stage approach is proposed to learning from partial label examples based on *feature-aware disambiguation*. In the first stage, the manifold structure of feature space is utilized to generate normalized labeling confidences over candidate label set. In the second stage, the predictive model is learned by performing regularized multi-output regression over the generated labeling confidences. Extensive experiments on artificial as well as real-world partial label data sets clearly validate the superiority of the proposed feature-aware disambiguation approach.

Keywords

weak supervision; partial label learning; disambiguation; manifold

1. INTRODUCTION

Partial label (PL) learning refers to the problem where each training example is represented by a single instance (feature vector) while associated with a set of *candidate* labels [8, 26]. Among the candidate label set, only one label is assumed to be valid and not directly accessible to the learning algorithm. The need to learn from data with partial labeling information naturally arises in many real-world applications such as automatic image annotation [7, 25], web mining [14], ecoinformatics [16], etc.[1]

[1]In some literatures, the partial label learning framework is also named as *ambiguous label learning* [5, 13] or *superset label learning* [17].

KDD '16, August 13-17, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939672.2939788

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ denote the d-dimensional feature space and $\mathcal{Y} = \{y_1, y_2, \ldots, y_q\}$ denote the label space with q class labels. The task of partial label learning is to learn a *multi-class* classifier $f : \mathcal{X} \mapsto \mathcal{Y}$ from the partial label training set $\mathcal{D} = \{(\boldsymbol{x}_i, S_i) \mid 1 \leq i \leq m\}$. Here, for each PL training example $(\boldsymbol{x}_i, S_i)$, $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})^\top$ is a d-dimensional feature vector and $S_i \subseteq \mathcal{Y}$ is the set of candidate labels associated with $\boldsymbol{x}_i$. Partial label learning takes the basic assumption that the ground-truth label y_i for $\boldsymbol{x}_i$ resides in its candidate label set (i.e. $y_i \in S_i$) but unknown to the learning algorithm.

Apparently, the available labeling information in the training set is ambiguous as the ground-truth label is concealed in the candidate label set. The key for successful partial label learning is therefore trying to disambiguate the set of candidate labels, where existing strategies include disambiguation by identification or disambiguation by averaging. For identification-based disambiguation, the ground-truth label is regarded as latent variable and identified through iterative refining procedure such as EM [5, 15, 16, 18, 24]. For averaging-based disambiguation, all the candidate labels are treated equally and the prediction is made by averaging their modeling outputs [8, 13, 27].

By taking specific views on the candidate labels, both of the existing strategies conduct disambiguation by only focusing on the manipulation of label space. Nonetheless, it is natural to postulate that the potentially useful information from feature space should also be exploited to facilitating the disambiguation process. Specifically, to help disambiguate the candidate label set, one might make 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. For instance, suppose we have three PL training examples $(\boldsymbol{x}, \{y_2, y_3\})$, $(\boldsymbol{x}', \{y_1, y_2\})$ and $(\boldsymbol{x}'', \{y_3, y_4\})$ where $\boldsymbol{x}$ is shown to be close to $\boldsymbol{x}'$ while far from $\boldsymbol{x}''$ in the feature space. Then, it is reasonable to assign higher labeling confidence on y_2 than y_3 for $\boldsymbol{x}$ (i.e. $\mathbb{P}(y_2 \mid \boldsymbol{x}) > \mathbb{P}(y_3 \mid \boldsymbol{x})$), as y_2 is a shared candidate label between close instances ($\boldsymbol{x}$ and $\boldsymbol{x}'$) while y_3 is a shared candidate label between distant instances ($\boldsymbol{x}$ and $\boldsymbol{x}''$).

In light of the above observation, a novel two-stage approach named PL-LEAF, i.e. *Partial Label LEArning via Feature-aware disambiguation*, is proposed in this paper. In the first stage, the manifold structure among training examples is analyzed in the feature space and then utilized to generate normalized labeling confidences over candidate label set. In the second stage, a multi-class predictive model is learned by fitting a regularized multi-output regressor with the generated labeling confidences. Extensive experiments on controlled UCI data sets as well as real-world PL data sets clearly show the effectiveness of feature-aware disambiguation for partial label learning.

The rest of this paper is organized as follows. Section 2 briefly

reviews related work in partial label learning. Section 3 introduces the proposed PL-LEAF approach. Section 4 reports experimental results of comparative studies. Finally, Section 5 concludes the paper and discusses future research issues.

2. RELATED WORK

Due to the ambiguous labeling information conveyed by PL training examples, partial label learning can be regarded as a *weakly-supervised* learning framework. It situates between two ends of the supervision spectrum, i.e. standard supervised learning with explicit supervision and unsupervised learning with blind supervision. Furthermore, partial label learning is related to other well-studied weakly-supervised learning frameworks, including *semi-supervised learning*, *multi-instance learning* and *multi-label learning*, while the weak supervision scenario for partial label learning is different to those counterpart frameworks.

Semi-supervised learning [4, 29] aims to learn a predictive model $f : \mathcal{X} \mapsto \mathcal{Y}$ from few labeled data together with abundant unlabeled data. For unlabeled data the ground-truth label assumes the entire label space, while for PL data the ground-truth label is confined within its candidate label set. Multi-instance learning [1, 9] aims to learn a predictive model $f : 2^{\mathcal{X}} \mapsto \mathcal{Y}$ from training examples each represented as a labeled bag of instances. For multi-instance data the label is assigned to bag of instances, while for PL data the label is assigned to single instance. Multi-label learning [11, 28] aims to learn a predictive model $f : \mathcal{X} \mapsto 2^{\mathcal{Y}}$ from training examples each associated with multiple labels. For multi-label data the associated labels are all valid ones, while for PL data the associated labels are only candidate ones.

Existing approaches learn from PL training examples by trying to disambiguate their candidate label sets. One disambiguation strategy is to assume certain parametric model $F(\boldsymbol{x}, y; \boldsymbol{\theta})$ where the ground-truth label is regarded as latent variable and identified as $\hat{y}_i = \arg\max_{y \in \mathcal{S}_i} F(\boldsymbol{x}_i, y; \boldsymbol{\theta})$. Generally, the latent variable is refined iteratively via EM procedure which optimizes objective function defined according to the maximum likelihood criterion: $\sum_{i=1}^{m} \log \left(\sum_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\theta}) \right)$ [15, 16], or the maximum margin criterion: $\sum_{i=1}^{m} (\max_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\theta}) - \max_{y \notin S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\theta}))$ [18, 24]. One potential drawback of the identification-based disambiguation strategy lies in that, rather than recovering the ground-truth label y_i, the identified label $\hat{y}_i$ might turn out to be false positive label in the candidate label set (i.e. $S_i \setminus \{y_i\}$).

Another disambiguation strategy is to assume equal importance of each candidate label and then make prediction by averaging their modeling outputs. Under discriminative learning setting, the averaged output from all candidate labels, i.e. $\frac{1}{|S_i|} \sum_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\theta})$, is distinguished from the outputs from non-candidate labels, i.e. $F(\boldsymbol{x}_i, y; \boldsymbol{\theta})$ $(y \notin S_i)$ [8]. Under instance-based learning setting, the predicted label for unseen instance is determined by averaging the candidate labeling information from its neighboring examples in the PL training set [13, 27]. One potential drawback of the averaging-based disambiguation strategy lies in that the essential modeling output from ground-truth label y_i might be overwhelmed by the distractive outputs from false positive labels.

In the next section, a novel partial label learning approach named PL-LEAF will be introduced. Other than disambiguation by identification or averaging, PL-LEAF facilitates the disambiguation process by making use of local topological information from the feature space. The candidate label set is disambiguated in the form of normalized labeling confidences, which differs from the crisp-style disambiguation of identifying a single candidate label or the uniform-style disambiguation of averaging all candidate labels.

3. THE PROPOSED APPROACH

As shown in Section 1, the task of partial label learning is to learn a multi-class classifier $f : \mathcal{X} \mapsto \mathcal{Y}$ from the PL training set $\mathcal{D} = \{(\boldsymbol{x}_i, S_i) \mid 1 \leq i \leq m\}$. For the proposed PL-LEAF approach, its key novelty lies in how the disambiguation procedure is conducted. Specifically, for each PL training example $(\boldsymbol{x}_i, S_i)$, PL-LEAF aims to disambiguate its candidate label set S_i via a normalized real-valued vector $\boldsymbol{\lambda}_i = (\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{iq})^\top$. Here, each component λ_{ik} $(1 \leq k \leq q)$ represents the *labeling confidence* of y_k being the ground-truth label for $\boldsymbol{x}_i$, which satisfies the following constraints:

- $\lambda_{ik} = 0 \ \ (\forall\, y_k \notin S_i)$
- $\lambda_{ik} \geq 0 \ \ (\forall\, y_k \in S_i)$ and $\sum_{y_k \in S_i} \lambda_{ik} = 1$

Once the normalized labeling confidence vectors have been generated, the predictive model will be induced by utilizing the disambiguation results.

In the next subsections, the two basic stages of PL-LEAF, i.e. *feature-aware disambiguation* and *predictive model induction*, will be scrutinized respectively.

3.1 Feature-Aware Disambiguation

In the first stage, PL-LEAF aims to disambiguate the candidate label set by exploiting useful information from the feature space. To fulfill this task, the popular smoothness assumption is adopted to enable information exploitation from the feature space to the label space. Given the PL training set $\mathcal{D} = \{(\boldsymbol{x}_i, S_i) \mid 1 \leq i \leq m\}$, a weighted graph $\mathcal{G} = (V, E, \mathbf{W})$ is constructed over the training examples to characterize the manifold structure of feature space. Here, $V = \{\boldsymbol{x}_i \mid 1 \leq i \leq m\}$ corresponds to the set of vertices and $E = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \boldsymbol{x}_i \in \mathrm{KNN}(\boldsymbol{x}_j),\ i \neq j\}$ corresponds to the set of directed edges from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ iff $\boldsymbol{x}_i$ is among the K-nearest neighbors of $\boldsymbol{x}_j$. Furthermore, $\mathbf{W} = [W_{ij}]_{m \times m}$ corresponds to the non-negative weight matrix where $W_{ij} = 0$ if $(\boldsymbol{x}_i, \boldsymbol{x}_j) \notin E$. For the weight matrix, its j-th column $\mathbf{W}_{\cdot j} = (W_{1j}, W_{2j}, \ldots, W_{mj})^\top$ is determined by solving the following linear least square problem:

$$\begin{aligned} \min_{\mathbf{W}_{\cdot j}} \ & \left\| \boldsymbol{x}_j - \textstyle\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in E} W_{ij} \cdot \boldsymbol{x}_i \right\|^2 \\ \text{s.t.:} \ & \textstyle\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in E} W_{ij} = 1 \\ & W_{ij} \geq 0 \ \ (\forall\, (\boldsymbol{x}_i, \boldsymbol{x}_j) \in E) \end{aligned} \tag{1}$$

Conceptually, the weight value W_{ij} $((\boldsymbol{x}_i, \boldsymbol{x}_j) \in E)$ characterizes the relative importance of neighboring example $\boldsymbol{x}_i$ in reconstructing $\boldsymbol{x}_j$. According to the constraint $\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in E} W_{ij} = 1$, the above optimization problem (**OP**) can be re-written as:

$$\begin{aligned} \min_{\mathbf{W}_{\cdot j}} \ & \mathbf{W}_{\cdot j}^\top \, \mathbf{G}^j \, \mathbf{W}_{\cdot j} \\ \text{s.t.:} \ & \mathbf{1}^\top \, \mathbf{W}_{\cdot j} = 1 \\ & W_{ij} \geq 0 \ \ (\forall\, (\boldsymbol{x}_i, \boldsymbol{x}_j) \in E) \\ & W_{ij} = 0 \ \ (\forall\, (\boldsymbol{x}_i, \boldsymbol{x}_j) \notin E) \end{aligned} \tag{2}$$

Here, $\mathbf{G}^j = [G^j_{ab}]_{m \times m}$ is the local Gram matrix for $\boldsymbol{x}_j$ with elements $G^j_{ab} = (\boldsymbol{x}_j - \boldsymbol{x}_a)^\top (\boldsymbol{x}_j - \boldsymbol{x}_b)$. Apparently, **OP**(2) corresponds to a standard quadratic programming (QP) problem whose optimal solution can be obtained by any off-the-shelf QP solver. The weight matrix $\mathbf{W}$ is constructed by solving **OP**(2) column-wisely and the resulting matrix is generally not symmetric. Accordingly, based on the local topological information embodied in $\mathcal{G}$, the manifold structure in the feature space will be exploited to help disambiguate the candidate label set.

Here, in order to generate the labeling confidence vectors $\boldsymbol{\Lambda} = [\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots, \boldsymbol{\lambda}_m]$, PL-LEAF exploits the smoothness assumption

Table 1: The pseudo-code of PL-LEAF.

Inputs:

$\mathcal{D}$: the partial label training set $\{(\boldsymbol{x}_i, S_i) \mid 1 \le i \le m\}$ $(\boldsymbol{x}_i \in \mathcal{X}, S_i \subseteq \mathcal{Y}, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{y_1, y_2, \ldots, y_q\})$

K: the number nearest neighbors used for weighted graph construction

C_1, C_2: the regularization parameters for regression loss function

$\boldsymbol{x}$: the unseen instance ($\boldsymbol{x} \in \mathcal{X}$)

Outputs:

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

Process:

Set the weighted graph $\mathcal{G} = (V, E, \mathbf{W})$ with $V = \{\boldsymbol{x}_i \mid 1 \le i \le m\}$ and $E = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \boldsymbol{x}_i \in \mathrm{KNN}(\boldsymbol{x}_j),\ i \neq j\}$;
for $j = 1$ **to** m **do**
  Instantiate the j-th column $\mathbf{W}_{\cdot j}$ of the weight matrix $\mathbf{W}$ by solving **OP**(2) with QP procedure;
end for
Generate the labeling confidence vectors $\boldsymbol{\Lambda}$ by solving **OP**(4) with QP procedure, or by solving **OP**(5) with alternating optimization;
Calculate the kernel matrix $\mathbf{K} = [\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{m \times m}$ over training examples;
Calculate $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_m]$ with $\boldsymbol{\psi}_i = -\left(\frac{1}{|S_i|} \cdot \mathbf{1}_{S_i} - \frac{1}{|\hat{S}_i|} \cdot \mathbf{1}_{\hat{S}_i}\right)$ $(1 \le i \le m)$
Set $t = 0$;
Initialize $\boldsymbol{\Theta}^{(0)}$ and $\boldsymbol{b}^{(0)}$ with $\boldsymbol{\alpha}_k^{(0)} = \mathbf{0}$ and $b_k^{(0)} = 0$ $(1 \le k \le q)$;
repeat
  Calculate $\boldsymbol{\rho} = [\rho_1, \rho_2, \ldots, \rho_m]^\top$ according to Eq.(11), and set $\mathbf{D}_\rho = [d_{ij}]_{m \times m}$ with $d_{ij} = \rho_i \delta_{ij}$;
  for $k = 1$ **to** q **do**
    Obtain the solution $\tilde{\boldsymbol{\alpha}}_k$ and $\tilde{b}_k$ by solving Eq.(17);
  end for
  Set the descending direction $\mathbf{P}^{(t)}$ according to Eq.(16);
  Update $\{\boldsymbol{\Theta}^{(t+1)}, \boldsymbol{b}^{(t+1)}\}$ by invoking line search procedure from $\{\boldsymbol{\Theta}^{(t)}, \boldsymbol{b}^{(t)}\}$ along $\mathbf{P}^{(t)}$;
  $t = t + 1$;
until $L(\boldsymbol{\Theta}^{(t-1)}, \boldsymbol{b}^{(t-1)}) - L(\boldsymbol{\Theta}^{(t)}, \boldsymbol{b}^{(t)}) < \tau \cdot L(\boldsymbol{\Theta}^{(t-1)}, \boldsymbol{b}^{(t-1)})$
Set the final predictive model with $\boldsymbol{\Theta}^* = \boldsymbol{\Theta}^{(t)}$ and $\boldsymbol{b}^* = \boldsymbol{b}^{(t)}$;
Return $y = f(\boldsymbol{x})$ according to Eq.(18).

that the manifold structure in the feature space should also be preserved in the label space:

$$\begin{aligned} \min_{\boldsymbol{\Lambda}} \quad & \sum_{j=1}^{m} \left\| \boldsymbol{\lambda}_j - \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in E} W_{ij} \cdot \boldsymbol{\lambda}_i \right\|^2 \\ \text{s.t.:} \quad & \lambda_{jk} = 0 \ \ (\forall\, 1 \le j \le m,\ y_k \notin S_j) \\ & \lambda_{jk} \ge 0 \ \ (\forall\, 1 \le j \le m,\ y_k \in S_j) \\ & \sum_{y_k \in S_j} \lambda_{jk} = 1 \ \ (\forall\, 1 \le j \le m) \end{aligned} \tag{3}$$

According to the constraint $\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in E} W_{ij} = 1$, the above **OP**(3) can be re-written as:

$$\begin{aligned} \min_{\boldsymbol{\Lambda}} \quad & \sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij} \cdot \boldsymbol{\lambda}_i^\top \boldsymbol{\lambda}_j \\ \text{s.t.:} \quad & \lambda_{jk} = 0 \ \ (\forall\, 1 \le j \le m,\ y_k \notin S_j) \\ & \lambda_{jk} \ge 0 \ \ (\forall\, 1 \le j \le m,\ y_k \in S_j) \\ & \sum_{y_k \in S_j} \lambda_{jk} = 1 \ \ (\forall\, 1 \le j \le m) \end{aligned} \tag{4}$$

Here, $\gamma_{ij} = \mathbf{W}_{i\cdot}\, \mathbf{W}_{j\cdot}^\top + \delta_{ij} \mathbf{W}_{\cdot i}^\top\, \mathbf{1}_{m \times m}\, \mathbf{W}_{\cdot i} - 2W_{ij}$ where $\mathbf{W}_{i\cdot}$, δ_{ij} and $\mathbf{1}_{m \times m}$ represent the i-th row of weight matrix $\mathbf{W}$, the Kronecker's delta and the $m \times m$ matrix of 1's respectively.

Note that **OP**(4) corresponds to a standard QP problem with mq variables and $m(q+1)$ constraints, whose computational complexity would be demanding when mq is large. To improve efficiency, we can choose to solve **OP**(4) with *alternating optimization* strategy where a series of QP subproblems with q variables and $q+1$ constraints are optimized iteratively. Without loss of generality, in each alternating optimization iteration, one labeling confidence vector $\boldsymbol{\lambda}_i$ is optimized by fixing the values of other labeling confidence vectors $\boldsymbol{\lambda}_j$ $(j \neq i)$:

$$\begin{aligned} \min_{\boldsymbol{\lambda}_i} \quad & r_{ii} \cdot \boldsymbol{\lambda}_i^\top \boldsymbol{\lambda}_i + \left(\sum_{j=1, j \neq i}^{m} (r_{ij} + r_{ji}) \cdot \boldsymbol{\lambda}_j^\top \right) \boldsymbol{\lambda}_i \\ \text{s.t.:} \quad & \lambda_{ik} = 0 \ \ (\forall\, y_k \notin S_i) \\ & \lambda_{ik} \ge 0 \ \ (\forall\, y_k \in S_i) \\ & \sum_{y_k \in S_i} \lambda_{ik} = 1 \end{aligned} \tag{5}$$

It is interesting to notice that, to some extent, the disambiguation results returned by existing strategies can be viewed as simplified versions of PL-LEAF's normalized labeling confidence vectors. For identification-based disambiguation [16, 18], a single label $y_{\hat{k}}$ in S_i is identified as the ground-truth label leading to *unimodal* labeling confidence vector with $\lambda_{i\hat{k}} = 1$ and $\lambda_{ik} = 0$ $(k \neq \hat{k})$. For averaging-based disambiguation [8, 13], all candidate labels in S_i are treated equally leading to *uniform* labeling confidence vector with $\lambda_{ik} = \frac{1}{|S_i|}$ $(y_k \in S_i)$. Therefore, the normalized labeling confidence vector considered by PL-LEAF would accommodate

Table 2: Characteristics of the experimental data sets.

Controlled UCI Data Sets				Configurations
Data set	# Examples	# Features	# Class Labels	
vehicle	846	18	4	(I) $p = 1, r = 1, \varepsilon \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 1]**
segment	2,310	18	7	(II) $r = 1, p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 2]**
abalone	4,177	7	29	(III) $r = 2, p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 3]**
satimage	6,345	36	7	(IV) $r = 3, p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 4]**
usps	9,298	256	10	
pendigits	10,992	16	10	

Real-World Data Sets					
Data set	# Examples	# Features	# Class Labels	Avg. # CLs	Task Domain
FG-NET	1,002	262	78	7.48	*facial age estimation* [20]
Lost	1,122	108	16	2.23	*automatic face naming* [8]
MSRCv2	1,758	48	23	3.16	*object classification* [16]
BirdSong	4,998	38	13	2.18	*bird song classification* [3]
Soccer Player	17,472	279	171	2.09	*automatic face naming* [25]
Yahoo! News	22,991	163	219	1.91	*automatic face naming* [12]

more flexibility in modeling the disambiguation results compared to the unimodal or uniform setup.

3.2 Predictive Model Induction

Following the first stage of feature-aware disambiguation, the original PL training set $\mathcal{D}$ has been transformed into its disambiguated counterpart: $\mathcal{D}_{\text{dis}} = \{(\boldsymbol{x}_i, \boldsymbol{\lambda}_i) \mid 1 \leq i \leq m\}$. In the second stage, PL-LEAF aims to induce the predictive model $f : \mathcal{X} \mapsto \mathcal{Y}$ based on $\mathcal{D}_{\text{dis}}$. Considering that the response variables (normalized labeling confidences) for each training example in $\mathcal{D}_{\text{dis}}$ are actually real-valued, it is natural to induce the predictive model by performing *multi-output regression*. Among various choices of multi-output regression techniques, we choose to adapt the multi-regression support vector machines (MSVR) [6, 21, 23] such that kernel trick can be readily incorporated to accommodate nonlinear modeling.

Let $\phi(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}^{\mathcal{H}_\kappa}$ be the (implicit) nonlinear mapping from the original feature space to the higher-dimensional Reproducing Kernel Hilbert Space (RKHS) via kernel function $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$. Furthermore, let $\{(\boldsymbol{\theta}_k, b_k) \mid 1 \leq k \leq q\}$ denote the multi-output regression model in the RKHS with one linear predictor $(\boldsymbol{\theta}_k, b_k)$ for each class label $y_k \in \mathcal{Y}$. Then, PL-LEAF induces the regression model by minimizing the following loss function:

$$L(\boldsymbol{\Theta}, \boldsymbol{b}) = \frac{1}{2}\sum_{k=1}^{q} ||\boldsymbol{\theta}_k||^2 + C_1 \sum_{i=1}^{m} L_1(u_i) + C_2 \sum_{i=1}^{m} v_i \qquad (6)$$

Here, $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_q]$ and $\boldsymbol{b} = [b_1, b_2, \ldots, b_q]^\top$ represent the regression model's weight matrix and bias vector respectively.

As shown in Eq.(6), the first term of $L(\boldsymbol{\Theta}, \boldsymbol{b})$ controls the complexity of the induced model. In addition, the second term of $L(\boldsymbol{\Theta}, \boldsymbol{b})$ is defined based on the ϵ-insensitive loss function:

$$L_1(u) = \begin{cases} 0, & u < \epsilon \\ (u - \epsilon)^2, & u \geq \epsilon \end{cases} \qquad (7)$$

For each example $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$ in $\mathcal{D}_{\text{dis}}$, the corresponding input to the ϵ-insensitive loss function $L_1(\cdot)$ is set as: $u_i = ||\boldsymbol{e}_i|| = \sqrt{\boldsymbol{e}_i^\top \boldsymbol{e}_i}$ with $\boldsymbol{e}_i = \boldsymbol{\lambda}_i - \boldsymbol{\Theta}^\top \phi(\boldsymbol{x}_i) - \boldsymbol{b}$. In this way, the outputs of all linear predictors are considered simultaneously to yield a unique input to $L_1(\cdot)$ such that the dependencies among all the class labels can be exploited by the ϵ-insensitive term. The third term of $L(\boldsymbol{\Theta}, \boldsymbol{b})$ considers the partial label loss for each example which is set as:

$$v_i = -\left(\frac{1}{|S_i|} \cdot \mathbf{1}_{S_i}^\top - \frac{1}{|\hat{S}_i|} \cdot \mathbf{1}_{\hat{S}_i}^\top\right)\left(\boldsymbol{\Theta}^\top \phi(\boldsymbol{x}_i) + \boldsymbol{b}\right) \qquad (8)$$

Here, for candidate label set S_i and its complementary set $\hat{S}_i$ in $\mathcal{Y}$, $\mathbf{1}_{S_i}$ ($\mathbf{1}_{\hat{S}_i}$) corresponds to a q-dimensional vector whose k-th element equals to 1 if $y_k \in S_i$ ($y_k \in \hat{S}_i$) and 0 otherwise. In other words, the third term enforces the property that the average output from candidate labels should be larger than the average output from non-candidate ones, which has been widely-used in designing effective partial label learning algorithms [8, 13, 26].

To minimize $L(\boldsymbol{\Theta}, \boldsymbol{b})$, PL-LEAF employs the gradient-based iterative method named Iterative Re-Weighted Least Square (IRWLS) [21, 23]. Specifically, in each iteration the descending direction is determined analytically by solving linear systems of equations. Let $\{\boldsymbol{\Theta}^{(t)}, \boldsymbol{b}^{(t)}\}$ denote the current model after t-th iteration, the ϵ-insensitive function $L_1(u_i)$ is firstly approximated by its first-order Taylor expansion:

$$\bar{L}_1(u_i) = L_1(u_i^{(t)}) + \left.\frac{dL_1(u)}{du}\right|_{u_i^{(t)}} \frac{(\boldsymbol{e}_i^{(t)})^\top}{u_i^{(t)}}\left(\boldsymbol{e}_i - \boldsymbol{e}_i^{(t)}\right) \qquad (9)$$

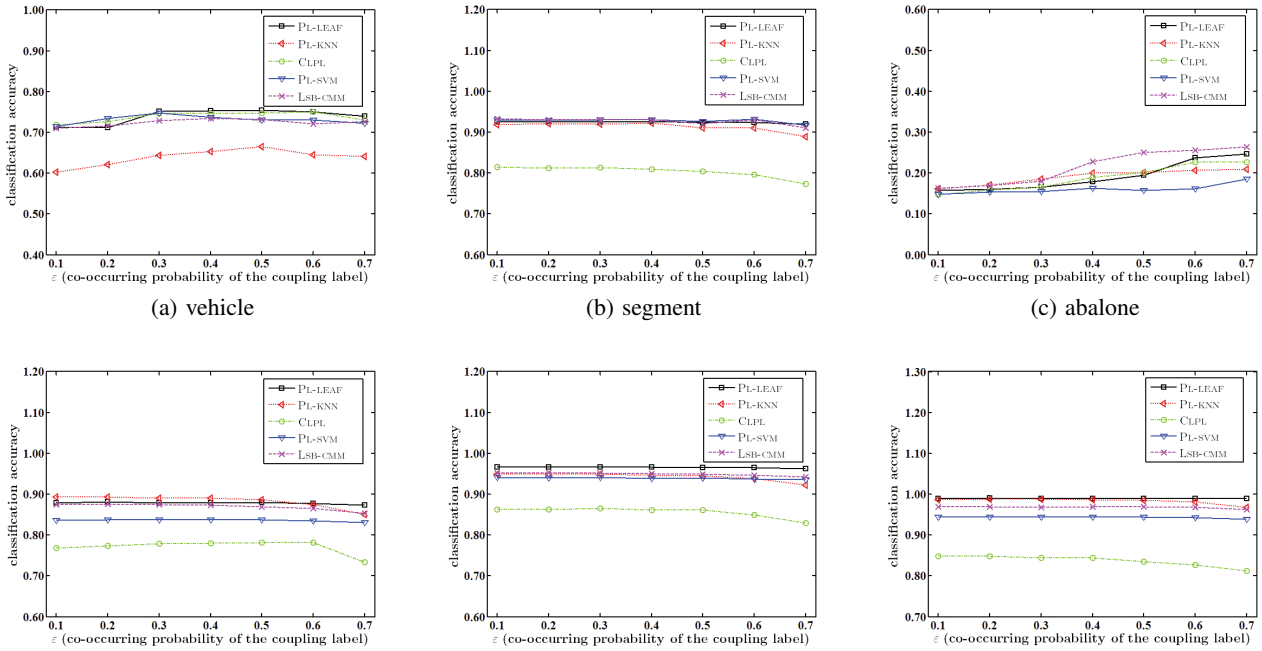
Here, $u_i^{(t)}$ and $\boldsymbol{e}_i^{(t)}$ are calculated based on the current model $\{\boldsymbol{\Theta}^{(t)}, \boldsymbol{b}^{(t)}\}$. Then, a quadratic approximation to $\bar{L}(u_i)$ is further constructed to ease analytical solution to the descending direction:

$$\begin{aligned} \tilde{L}_1(u_i) &= L_1(u_i^{(t)}) + \left.\frac{dL_1(u)}{du}\right|_{u_i^{(t)}} \frac{u_i^2 - (u_i^{(t)})^2}{2u_i^{(t)}} \qquad (10) \\ &= \frac{1}{2}\rho_i u_i^2 + T_i \end{aligned}$$

where

$$\rho_i = \frac{1}{u_i^{(t)}} \left.\frac{L_1(u)}{du}\right|_{u_i^{(t)}} = \begin{cases} 0, & u_i^{(t)} < \epsilon \\ \frac{2\left(u_i^{(t)} - \epsilon\right)}{u_i^{(t)}}, & u_i^{(t)} \geq \epsilon \end{cases} \qquad (11)$$

and T_i is a constant which does not depend on $\{\boldsymbol{\Theta}, \boldsymbol{b}\}$.



(a) vehicle (b) segment (c) abalone

(d) satimage (e) usps (f) pendigits

Figure 1: Classification accuracy of each comparing algorithm changes as ε (co-occurring probability of the coupling label) increases from 0.1 to 0.7 (with 100% partially labeled examples [$p = 1$] and one false positive candidate label [$r = 1$]).

Based on Eqs.(10) and (11), the objective function $L(\boldsymbol{\Theta}, \boldsymbol{b})$ can be re-written as:

$$\tilde{L}(\boldsymbol{\Theta}, \boldsymbol{b}) = \frac{1}{2}\sum_{k=1}^{q}||\boldsymbol{\theta}_k||^2 + C_1\sum_{i=1}^{m}\frac{\rho_i}{2}u_i^2 + C_2\sum_{i=1}^{m}v_i + C_1\sum_{i=1}^{m}T_i \tag{12}$$

In contrast to standard least square objective function $\frac{1}{2}\sum_{k=1}^{q}||\boldsymbol{\theta}_k||^2 + C_1\sum_{i=1}^{m}u_i^2$, the minimization of Eq.(12) can be regarded as a *weighted* least square problem along with partial label loss regularization. Minimization of $\tilde{L}(\boldsymbol{\Theta}, \boldsymbol{b})$ can be decoupled for each class label, whose solution for each $(\boldsymbol{\theta}_k, b_k)$ $(1 \le k \le q)$ is found by equating the corresponding gradient to zero:

$$\nabla_{\boldsymbol{\theta}_k}\tilde{L}(\boldsymbol{\Theta}, \boldsymbol{b}) = \boldsymbol{\theta}_k - C_1\sum_{i=1}^{m}\rho_i\phi(\boldsymbol{x}_i)(\lambda_{ik} - \phi(\boldsymbol{x}_i)^\top\boldsymbol{\theta}_k - b_k) + C_2\sum_{i=1}^{m}\psi_{ik}\phi(\boldsymbol{x}_i) = \mathbf{0} \tag{13}$$

$$\nabla_{b_k}\tilde{L}(\boldsymbol{\Theta}, \boldsymbol{b}) = -C_1\sum_{i=1}^{m}\rho_i(\lambda_{ik} - \phi(\boldsymbol{x}_i)^\top\boldsymbol{\theta}_k - b_k) + C_2\sum_{i=1}^{m}\psi_{ik} = 0 \tag{14}$$

Here, ψ_{ik} corresponds to the k-th component of the q-dimensional vector $\boldsymbol{\psi}_i = -\left(\frac{1}{|S_i|}\cdot\mathbf{1}_{S_i} - \frac{1}{|\bar{S}_i|}\cdot\mathbf{1}_{\bar{S}_i}\right)$. Accordingly, Eqs.(13) and (14) can be expressed as a linear system of equations:

$$\begin{bmatrix} C_1\boldsymbol{\Phi}^\top\mathbf{D}_\rho\boldsymbol{\Phi} + \mathbf{I} & C_1\boldsymbol{\Phi}^\top\boldsymbol{\rho} \\ C_1\boldsymbol{\rho}^\top\boldsymbol{\Phi} & C_1\mathbf{1}^\top\boldsymbol{\rho} \end{bmatrix}\begin{bmatrix}\boldsymbol{\theta}_k \\ b_k\end{bmatrix} = \begin{bmatrix} C_1\boldsymbol{\Phi}^\top\mathbf{D}_\rho\boldsymbol{\lambda}^k - C_2\boldsymbol{\Phi}^\top\boldsymbol{\psi}^k \\ C_1\boldsymbol{\rho}^\top\boldsymbol{\lambda}^k - C_2\mathbf{1}^\top\boldsymbol{\psi}^k \end{bmatrix} \tag{15}$$

Here, $\boldsymbol{\Phi} = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_m)]^\top$, $\mathbf{D}_\rho = [d_{ij}]_{m\times m}$ with $d_{ij} = \rho_i\delta_{ij}$, $\boldsymbol{\rho} = [\rho_1, \rho_2, \ldots, \rho_m]^\top$, $\boldsymbol{\lambda}^k = [\lambda_{1k}, \lambda_{2k}, \ldots, \lambda_{mk}]^\top$, $\boldsymbol{\psi}^k = [\psi_{1k}, \psi_{2k}, \ldots, \psi_{mk}]^\top$.

Let $\tilde{\boldsymbol{\Theta}} = [\tilde{\boldsymbol{\theta}}_1, \tilde{\boldsymbol{\theta}}_2, \ldots, \tilde{\boldsymbol{\theta}}_q]$ and $\tilde{\boldsymbol{b}} = [\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_q]^\top$ be the solution obtained by solving Eq.(15) for each class label, the descending direction for the next iteration would be:

$$\mathbf{P}^{(t)} = \begin{bmatrix}\tilde{\boldsymbol{\Theta}} - \boldsymbol{\Theta}^{(t)} \\ (\tilde{\boldsymbol{b}} - \boldsymbol{b}^{(t)})^\top\end{bmatrix} \tag{16}$$

The subsequent model $\{\boldsymbol{\Theta}^{(t+1)}, \boldsymbol{b}^{(t+1)}\}$ is then updated by invoking line search procedure from $\{\boldsymbol{\Theta}^{(t)}, \boldsymbol{b}^{(t)}\}$ along the descending direction $\mathbf{P}^{(t)}$ [19].

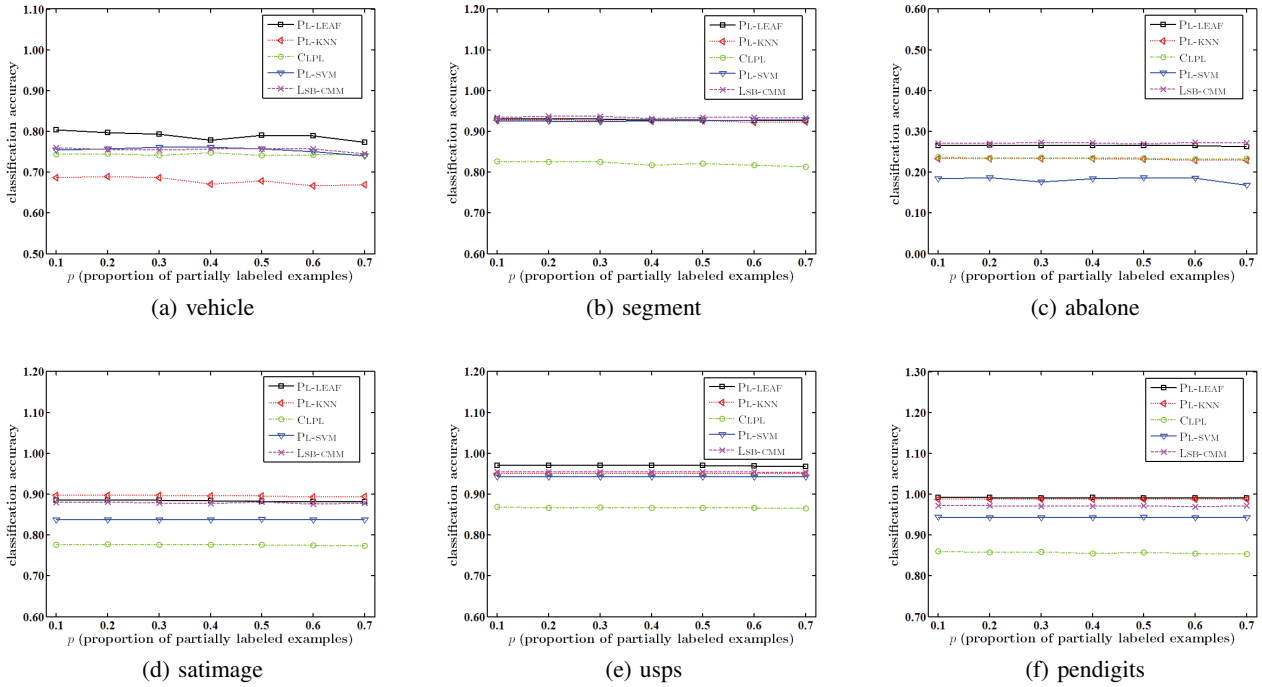
According to the Representer Theorem [22], under fairly general conditions, the predictive model can be expressed by a linear combination of the training examples in the RKHS, i.e. $\boldsymbol{\theta}_k = \sum_{i=1}^{m}\alpha_{ik}\phi(\boldsymbol{x}_i) = \boldsymbol{\Phi}^\top\boldsymbol{\alpha}_k$. By introducing the kernel trick into Eqs.(13) and (14), the linear system of Eq.(15) can be expressed as follows:

$$\begin{bmatrix} C_1\mathbf{K} + \mathbf{D}_\rho^{-1} & C_1\mathbf{1} \\ C_1\boldsymbol{\rho}^\top\mathbf{K} & C_1\mathbf{1}^\top\boldsymbol{\rho}\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_k \\ b_k\end{bmatrix} = \begin{bmatrix} C_1\boldsymbol{\lambda}^k - C_2\mathbf{D}_\rho^{-1}\boldsymbol{\psi}^k \\ C_1\boldsymbol{\rho}^\top\boldsymbol{\lambda}^k - C_2\mathbf{1}^\top\boldsymbol{\psi}^k\end{bmatrix} \tag{17}$$

Here, $\mathbf{K} = [\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{m\times m}$ is the kernel matrix over training examples. Based on the kernel trick, the line search procedure can be readily performed in terms of $\boldsymbol{\alpha}_k$ and b_k as well.

Let $\boldsymbol{\Theta}^*$ and $\boldsymbol{b}^*$ be the resulting model after the whole iterative optimization process, PL-LEAF makes prediction on the class label of unseen instance $\boldsymbol{x}$ as follows:

$$\begin{aligned} f(\boldsymbol{x}) &= \arg\max_{y_k\in\mathcal{Y}} \boldsymbol{\theta}_k^{*\top}\phi(\boldsymbol{x}) + b_k^* \\ &= \arg\max_{y_k\in\mathcal{Y}} \sum_{i=1}^{m}\alpha_{ik}^*\,\kappa(\boldsymbol{x}_i, \boldsymbol{x}) + b_k^* \end{aligned} \tag{18}$$



(a) vehicle (b) segment (c) abalone

(d) satimage (e) usps (f) pendigits

Figure 2: Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with one false positive candidate label $[r = 1]$).

Table 1 summarizes the pseudo-code of PL-LEAF.[2] Given the PL training set, a weighted graph is constructed to characterize the manifold structure of feature space which is then utilized to disambiguate the candidate label set (Steps 1-5). After that, a predictive model based on kernelized multiregression SVR is learned via gradient-based iterative optimization (Steps 6-18).[3] Finally, prediction on the unseen instance is made via the learned predictive model (Steps 19-20).

4. EXPERIMENTS

4.1 Experimental Setup

To evaluate the performance of PL-LEAF, two series of comparative experiments are conducted on controlled UCI data sets [2] as well as real-world partial label data sets. Characteristics of the experimental data sets are summarized in Table 2.

Following the widely-used controlling protocol in partial label learning research [5, 8, 16, 24, 27], an artificial PL data set can be generated from a multi-class UCI data set with three controlling parameters p, r and ε. Here, p controls the proportion of examples which are partially labeled (i.e. $|S_i| > 1$), r controls the number of false positive labels in the candidate label set (i.e. $|S_i| = r + 1$), and ε controls the co-occurring probability between one coupling candidate label and the ground-truth label. As shown in Table 2, a total of 28 (4x7) parameter configurations have been considered for each UCI data set.

In addition to artificial data sets, a number of real-world PL data sets have been collected from several task domains.[4] For the task of *facial age estimation* (`FG-NET` [20]), human faces with landmarks are represented as instances while ages annotated by ten crowd-sourced labelers together with the ground-truth age are regarded as candidate labels. For the task of *automatic face naming* (`Lost` [8], `Soccer Player` [25] and `Yahoo! News` [12]), faces cropped from an image or video frame are represented as instances while names extracted from the associated captions or subtitles are regarded as candidate labels. For the task of *bird song classification* (`BirdSong` [3]), 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 the task of *object classification* (`MSRCv2` [16]), image segmentations are represented as instances while objects appearing within the same image are regarded as candidate labels. The average number of candidate labels (Avg. #CLs) for each real-world PL data set is also recorded in Table 2.

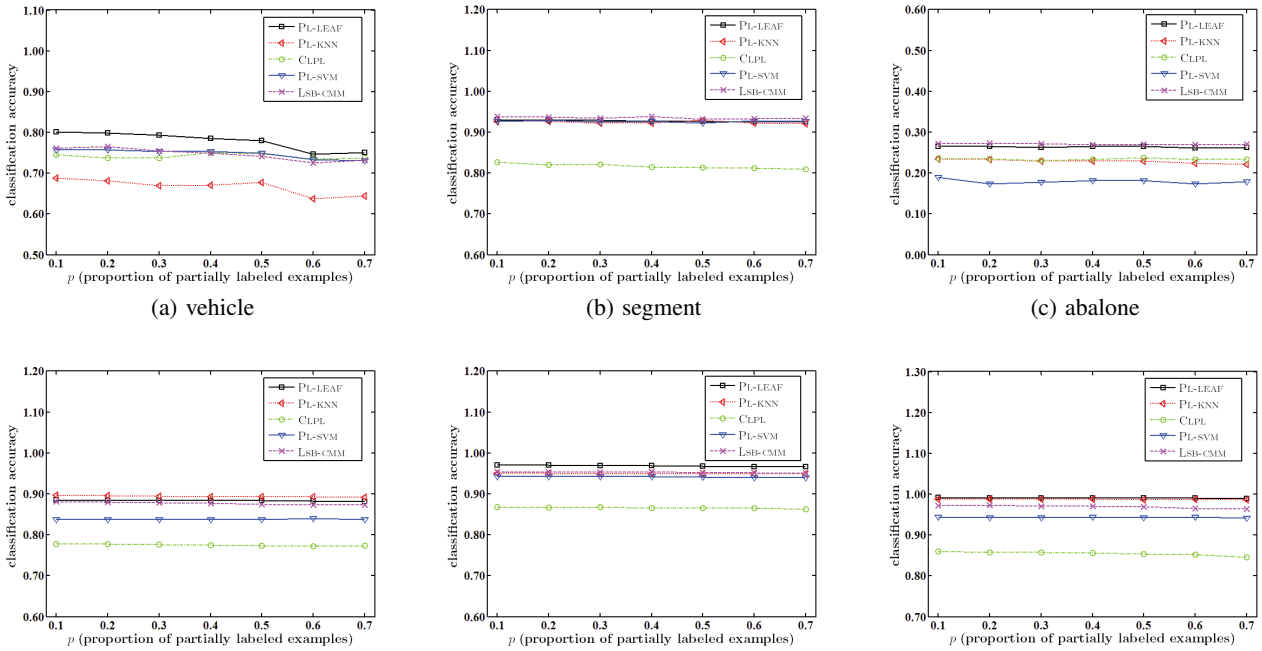
To show the effectiveness of feature-aware disambiguation, PL-LEAF is compared against four state-of-the-art partial label learning approaches with diverse properties, each configured with parameters suggested in respective literatures:

- PL-KNN [13]: A K-nearest neighbor approach to partial label learning via averaging-based disambiguation [suggested configuration: $K = 10$];
- CLPL [8]: A discriminative approach to partial label learning via averaging-based disambiguation [suggested configuration: SVM with squared hinge loss];
- PL-SVM [18]: A maximum-margin approach to partial label learning via identification-based disambiguation [suggested

[2]Code package for PL-LEAF is publicly-available at: http://cse.seu.edu.cn/PersonalPage/zhangml/Resources.htm#kdd16

[3]In this paper, the ϵ-insensitive function $L_1(\cdot)$ is instantiated with $\epsilon = 0.1$ and the convergence condition in Step 18 is instantiated with $\tau = 10^{-10}$.

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



(a) vehicle (b) segment (c) abalone

(d) satimage (e) usps (f) pendigits

Figure 3: Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with two false positive candidate labels $[r = 2]$).

configuration: regularization parameter pool with $\{10^{-3}, \ldots, 10^{3}\}$];

- LSB-CMM [16]: A maximum-likelihood approach to partial label learning via identification-based disambiguation [suggested configuration: q mixture components].

Parameters for PL-LEAF (Table 1) are set as $K = 10$, $C_1 = 10$ and $C_2 = 1$.[5] In this paper, ten runs of 50%/50% random train/test splits are performed on each artificial as well as real-world PL data set. Accordingly, the mean predictive accuracies (with standard deviation) are recorded for all comparing algorithms.

4.2 Experimental Results

4.2.1 *Controlled UCI Data Sets*

In Figure 1, the classification accuracy of each comparing algorithm is illustrated where the co-occurring probability ε varies from 0.1 to 0.7 with step-size 0.1 ($p = 1, r = 1$). For any ground-truth label $y \in \mathcal{Y}$, 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. Otherwise, any other class label would be chosen to co-occur with y. In Figures 2 to 4, the classification accuracy of each comparing algorithm is illustrated where the proportion p varies from 0.1 to 0.7 with step-size 0.1 ($r = 1, 2, 3$). Together with the ground-truth label, r class labels in $\mathcal{Y}$ will be randomly picked up to constitute the candidate label set for each partially labeled example.

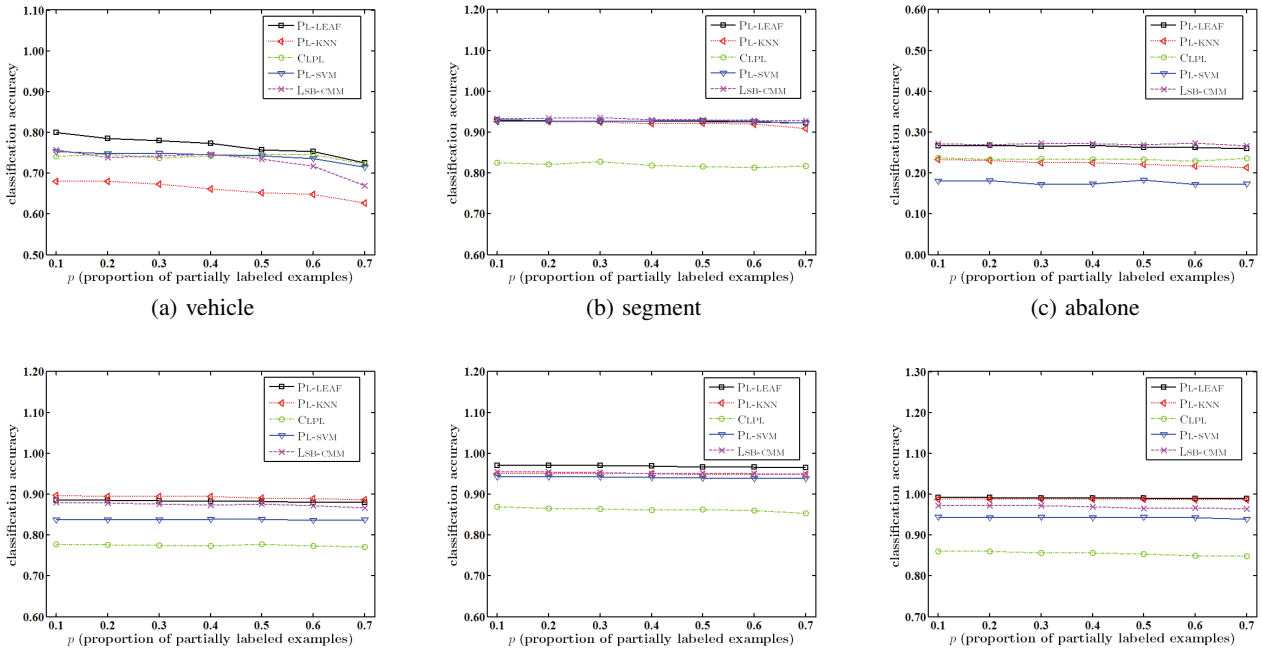
As shown in Figures 1 to 4, in most cases, PL-LEAF achieves superior or competitive performance against the comparing algorithms. Based on pairwise t-test at 0.05 significance level, Table 3 summarizes the win/tie/loss counts between PL-LEAF and each comparing algorithm. Out of the 168 statistical tests (28 configurations $\times$ 6 UCI data sets), it is shown that:

- Comparing to averaging-based disambiguation approaches, PL-LEAF achieves superior performance against PL-KNN and CLPL in 66.0% and 90.4% cases respectively. Furthermore, the performance of PL-LEAF is inferior to PL-KNN in only 17.9% cases and has never been outperformed by CLPL.
- Comparing to identification-based disambiguation approaches, PL-LEAF achieves superior performance against PL-SVM and LSB-CMM in 75.5% and 54.7% cases. Furthermore, the performance of PL-LEAF is inferior to LSB-CMM in only 11.9% cases and has been outperformed by PL-SVM in only 2 out of 168 cases.

Table 3: Win/tie/loss counts (pairwise t-test at 0.05 significance level) on the classification performance of PL-LEAF against each comparing algorithm.

	PL-LEAF **against**			
	PL-KNN	CLPL	PL-SVM	LSB-CMM
[Figure 1]	26/7/9	31/11/0	27/13/2	20/16/6
[Figure 2]	28/7/7	42/0/0	35/7/0	23/16/3
[Figure 3]	28/7/7	40/2/0	33/9/0	23/12/7
[Figure 4]	29/6/7	39/3/0	32/10/0	26/12/4
In Total	**111/27/30**	**152/16/0**	**127/39/2**	**92/56/20**

[5] For PL-LEAF, Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\sigma^2}\right)$ is employed in Step 6 with σ being the average distance among each pair of training examples. Furthermore, on the two artificial (`usps`, `pendigits`) and two real-world (`Soccer Player`, `Yahoo! News`) data sets with large scale, alternating optimization is employed in Step 5. Sensitivity analysis on PL-LEAF's parameter configuration is conducted in Subsection 4.3.



(a) vehicle (b) segment (c) abalone

(d) satimage (e) usps (f) pendigits

Figure 4: Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with three false positive candidate labels $[r = 3]$).

Table 4: Classification accuracy (mean±std) of each comparing algorithm on the real-world partial label data sets. In addition, •/○ indicates whether the performance of PL-LEAF is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significance level).

	PL-LEAF	PL-KNN	CLPL	PL-SVM	LSB-CMM
FG-NET	0.072±0.010	0.037±0.008•	0.047±0.017•	0.058±0.010•	0.056±0.008•
FG-NET (MAE3)	0.411±0.012	0.284±0.035•	0.240±0.045•	0.343±0.022•	0.344±0.026•
FG-NET (MAE5)	0.550±0.018	0.438±0.033•	0.343±0.055•	0.473±0.016•	0.478±0.025•
Lost	0.664±0.020	0.332±0.030•	0.670±0.024	0.639±0.056	0.591±0.019•
MSRCv2	0.459±0.013	0.417±0.012•	0.375±0.020•	0.417±0.027•	0.431±0.008•
BirdSong	0.706±0.012	0.637±0.009•	0.624±0.009•	0.671±0.018•	0.692±0.015•
Soccer Player	0.515±0.004	0.494±0.004•	0.347±0.004•	0.430±0.004•	0.506±0.006•
Yahoo! News	0.597±0.004	0.403±0.004•	0.457±0.005•	0.615±0.002○	0.594±0.007

4.2.2 Real-World Data Sets

Table 4 reports the detailed predictive performance of each comparing algorithm on the real-world PL data sets, where the outcomes of pairwise t-tests at 0.05 significance level are also recorded. Note that the average number of candidate labels (Avg. #CLs) for the FG-NET data set (i.e. 7.48 as shown in Table 2) is quite large, which makes the task of facial age estimation (based on training examples with partial labels) rather challenging. Furthermore, the state-of-the-art performance on this data set (based on training examples with ground-truth labels) corresponds to more than 3 years of mean average error (MAE) between the predicted age and the ground-truth age [20]. In Table 4, two extra classification accuracies are reported on the FG-NET data set where an unseen example is regarded to be correctly classified if the difference between the predicted age and the ground-truth age is less than 3 years (MAE3) or 5 years (MAE5).

As shown in Table 4, it is impressive to observe that:

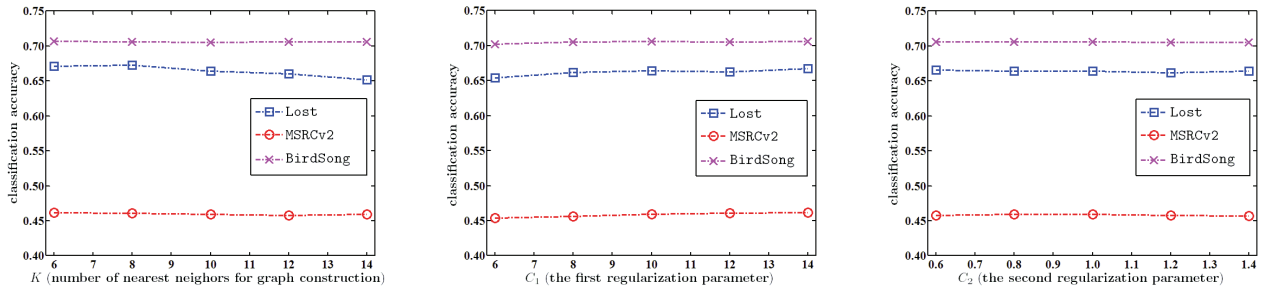
- On the FG-NET (with its MAE3 and MAE5 variants), MSRCv2, BirdSong and Soccer Player data sets, the performance of PL-LEAF is superior to all the other comparing algorithms.
- On the Lost data set, the performance of PL-LEAF is superior to PL-KNN and LSB-CMM, and comparable to CLPL and PL-SVM.
- On the Yahoo! News data set, the performance of PL-LEAF is superior to PL-KNN and CLPL, comparable to LSB-CMM, and inferior to PL-SVM.

4.3 Further Analysis

In addition to Table 4 reporting inductive performance on test examples, it is also interesting to study the *transductive* performance of each comparing algorithm on classifying training examples [8, 24]. For each PL training example $(\boldsymbol{x}_i, S_i)$, its 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. In other words, transductive

Table 5: Transductive accuracy (mean±std) of each comparing algorithm on the real-world partial label data sets. In addition, •/○ indicates whether the performance of PL-LEAF is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significance level).

	PL-LEAF	PL-KNN	CLPL	PL-SVM	LSB-CMM	PL-LEAF†
FG-NET	0.148±0.009	0.173±0.017○	0.158±0.018	0.136±0.021	0.138±0.019	0.142±0.010
FG-NET (MAE3)	0.567±0.015	0.559±0.023	0.548±0.017•	0.502±0.030•	0.541±0.022•	0.556±0.014
FG-NET (MAE5)	0.710±0.014	0.721±0.021	0.684±0.021•	0.644±0.021•	0.684±0.018•	0.698±0.010
Lost	0.809±0.022	0.596±0.018•	0.855±0.007○	0.814±0.033	0.755±0.018•	0.788±0.025
MSRCv2	0.645±0.015	0.603±0.015•	0.612±0.035•	0.656±0.026	0.603±0.016•	0.629±0.016
BirdSong	0.822±0.014	0.766±0.015•	0.820±0.014	0.831±0.013	0.827±0.017	0.787±0.016
Soccer Player	0.702±0.003	0.674±0.002•	0.742±0.005○	0.733±0.007○	0.688±0.003•	0.669±0.003
Yahoo! News	0.827±0.002	0.720±0.003•	0.832±0.002○	0.861±0.003○	0.861±0.002○	0.822±0.002



(a) Varying K ($C_1 = 10, C_2 = 1$) (b) Varying C_1 ($K = 10, C_2 = 1$) (c) Varying C_2 ($K = 10, C_1 = 10$)

Figure 5: Parameter sensitivity analysis for PL-LEAF on the `Lost`, `MSRCv2` and `BirdSong` data sets. (a) Classification accuracy of PL-LEAF changes as K increases from 6 to 14 with step-size 2; (b) Classification accuracy of PL-LEAF changes as C_1 increases from 6 to 14 with step-size 2; (c) Classification accuracy of PL-LEAF changes as C_2 increases from 0.6 to 1.4 with step-size 0.2.

performance of the partial label learning algorithm reflects its ability in disambiguating the candidate label set. Accordingly, Table 5 reports the transductive accuracy of each comparing algorithm together with the outcomes of pairwise t-tests at 0.05 significance level. Out of the 32 statistical tests (8 data sets × 4 comparing algorithm), it is shown that:

- Comparing to averaging-based disambiguation approaches, PL-LEAF is outperformed by PL-KNN on the `FG-NET` data set, and outperformed by CLPL on the `Lost`, `Soccer Player` and `Yahoo! News` data sets. In the rest 12 statistical tests, the performance of PL-LEAF is superior or at least comparable to PL-KNN and CLPL.

- Comparing to identification-based disambiguation approaches, PL-LEAF is outperformed by PL-SVM on the `Soccer Player` and `Yahoo! News` data sets, and outperformed by LSB-CMM on the `Yahoo! News` data set. In the rest 13 statistical tests, the performance of PL-LEAF is superior or at least comparable to PL-SVM and LSB-CMM.

As the feature-aware disambiguation stage of PL-LEAF finishes, the generated labeling confidence vector $\boldsymbol{\lambda}_i$ can also be used to predict the ground-truth label of $\boldsymbol{x}_i$ as $\hat{y}_i = \arg\max_{y_k \in S_i} \lambda_{ik}$. The resulting transductive performance is reported in Table 5 (denoted as PL-LEAF†) for referencee purpose. In most cases, the transductive performance of PL-LEAF† is close to PL-LEAF indicating the usefulness of generated labeling confidence vectors.

As shown in Table 1, to study the sensitivity of PL-LEAF w.r.t. its parameters K, C_1 and C_2, Figure 5 illustrates how PL-LEAF performs under different parameter configurations. For clarity of illustration, three data sets (`Lost`, `MSRCv2` and `BirdSong`) are chosen here for sensitivity analysis while similar observations also hold on other data sets.

It is obvious from Figure 5 that the performance of PL-LEAF is stable across a broad range of each parameter. This property is quite desirable as one can make use of PL-LEAF to achieve robust classification performance without the need of parameter fine-tuning. Therefore, the parameter configuration for PL-LEAF in Subsection 4.1 (K=10, C_1=10, C_2=1) naturally follows from these observations.

5. CONCLUSION

In this paper, a novel approach named PL-LEAF is proposed to learning from partial label examples. Different from existing strategies, PL-LEAF aims to disambiguate the candidate label set by manipulating useful information in the feature space. Specifically, PL-LEAF generates normalized labeling confidence vectors based on manifold relationships among training examples, and then induces the predictive model based on multi-output regression analysis. Comparative studies across comprehensive partial label data sets clearly verify the effectiveness of the proposed approach.

For PL-LEAF, an important future work is to investigate ways to perform manifold structure discovery and labeling confidence generation simultaneously. Secondly, it is worth studying whether there are better techniques to exploit the generated labeling confidence vectors, such as fitting probabilistic models [10]. Thirdly, it is also interesting to explore other ways to fulfill the feature-aware disambiguation strategy.

Acknowledgements

The authors wish to thank the anonymous reviewers for their constructive comments and suggestions. This work was supported by the National Science Foundation of China (61222309, 61573104, 61473087), the Natural Science Foundation of Jiangsu Province (BK20141340), the MOE Program for New Century Excellent Talents in University (NCET-13-0130), and the Collaborative Innovation Center of Wireless Communications Technology.

6. REFERENCES

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

[2] K. Bache and M. Lichman. UCI machine learning repository. School of Information and Computer Sciences, University of California, Irvine, 2013.

[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, Beijing, China, 2012.

[4] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

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

[6] W. Chung, J. Kim, H. Lee, and E. Kim. General dimensional multiple-output support vector regressions and their multiple kernel learning. *IEEE Transactions on Cybernetics*, 45(11):2572–2584, 2015.

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

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

[9] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[10] X. Geng, C. Yin, and Z.-H. Zhou. Facial age estimation by label distribution learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(10):2401–2412, 2013.

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

[12] M. Guillaumin, J. Verbeek, and C. Schmid. Multiple instance metric learning from automatically labeled bags of faces. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Lecture Notes in Computer Science 6311*, pages 634–647. Springer, Berlin, 2010.

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

[14] L. Jie and F. Orabona. Learning from candidate labeling sets. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1504–1512. MIT Press, Cambridge, MA, 2010.

[15] R. Jin and Z. Ghahramani. Learning with multiple labels. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 897–904. MIT Press, Cambridge, MA, 2003.

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

[17] L. Liu and T. Dietterich. Learnability of the superset label learning problem. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1629–1637, Beijing, China, 2014.

[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 381–389, Las Vegas, NV, 2008.

[19] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, Berlin, 2nd edition, 2006.

[20] G. Panis and A. Lanitis. An overview of research activities in facial age estimation using the FG-NET aging database. In L. Agapito, M. M. Bronstein, and C. Rother, editors, *Lecture Notes in Computer Science 8926*, pages 737–750. Springer, Berlin, 2015.

[21] M. Sánchez-Fernández, M. de Prado-Cumplido, J. Arenas-García, and F. Pérez-Cruz. SVM multiregression for nonliear channel estimation in multiple-input multiple-output systems. *IEEE Transactions on Signal Processing*, 52(8):2298–2307, 2004.

[22] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Camridge, MA, 2001.

[23] D. Tuia, J. Verrelst, L. Alonso, F. Pérez-Cruz, and G. Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, 8(4):804–808, 2011.

[24] F. Yu and M.-L. Zhang. Maximum margin partial label learning. In *Proceedings of the 7th Asian Conference on Machine Learning*, pages 96–111, Hong Kong, China, 2015.

[25] 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 Computer Society Conference on Computer Vision and Pattern Recognition*, pages 708–715, Portland, OR, 2013.

[26] M.-L. Zhang. Disambiguation-free partial label learning. In *Proceedings of the 14th SIAM International Conference on Data Mining*, pages 37–45, Philadelphia, PA, 2014.

[27] 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, Buenos Aires, Argentina, 2015.

[28] 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.

[29] X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. In R. J. Brachman and T. G. Dietterich, editors, *Synthesis Lectures to Artificial Intelligence and Machine Learning*, pages 1–130. Morgan & Claypool Publishers, San Francisco, CA, 2009.