Disambiguation-Free Partial Label Learning

Min-Ling Zhang, *Member, IEEE,* Fei Yu and Cai-Zhi Tang

Abstract—In partial label learning, each training example is associated with a set of *candidate* labels among which only one is the ground-truth label. The common strategy to induce predictive model is trying to disambiguate the candidate label set, i.e. differentiating the modeling outputs of individual candidate labels. Specifically, disambiguation by differentiation can be conducted either by identifying the ground-truth label iteratively or by treating each candidate label equally. Nonetheless, the disambiguation strategy is prone to be misled by the false positive labels co-occurring with ground-truth label. In this paper, a new partial label learning strategy is studied which refrains from conducting disambiguation. Specifically, by adapting error-correcting output codes (ECOC), a simple yet effective approach named PL-ECOC is proposed by utilizing candidate label set as *an entirety*. During training phase, to build binary classifier w.r.t. each column coding, any partially labeled example will be regarded as a positive or negative training example only if its candidate label set entirely falls into the coding dichotomy. During testing phase, class label for the unseen instance is determined via loss-based decoding which considers binary classifiers' empirical performance and predictive margin. Extensive experiments show that PL-ECOC performs favorably against state-of-the-art partial label learning approaches.

Index Terms—machine learning, partial label learning, disambiguation, weak supervision, error-correcting output codes

1 Introduction

Partial label (PL) learning deals with the problem where each training example is associated with a set of *candidate* labels, among which only one corresponds to the ground-truth label [14], [26]. In recent years, the need to learn from data with partial labels naturally arises in many real-world applications. For instance, in automatic face naming (Figure 1(a)), for a news document one can treat each face detected from the news picture as an instance and those names extracted from associated caption as candidate labels, while the actual correspondence between each face and its ground-truth label is not known [13], [35]; in online object annotation (Figure 1(b)), for a painting image one can treat web users' free annotations on its painting style as candidate labels, while the actual correspondence between the painting image and its ground-truth label is not known [25]. Successful applications of partial label learning techniques also include object classification [27], facial age estimation [34], [38], ecoinformatics [7], etc.

Reuters News: **Angela Merkel** and **Francois Hollande** joined **Vladimir Putin** in Moscow to discuss the escalating crisis in Ukraine

(a) automatic face naming

Annotation from user A: **Picasso style**

Annotation from user B: **Monet style**

Annotation from user C: **van Gogh style**

(b) online object annotation

Fig. 1. Exemplar applications of partial label learning. (a) Candidate names can be automatically extracted from the news caption, while the actual correspondence between each face and its ground-truth name is unknown [13], [35]; (b) Candidate annotations on the painting style can be freely provided by web users, while the actual correspondence between the painting image and its ground-truth annotation is unknown [25].

Formally, let $\mathcal{X} = \mathbb{R}^d$ be the d-dimensional instance space and $\mathcal{Y} = \{y_1, y_2, \ldots, y_q\}$ be the label space with q class labels. In addition, let $\mathcal{D} = \{(\boldsymbol{x}_i, S_i) \mid 1 \leq i \leq m\}$ be the training set consisting of m PL training examples, where each instance $\boldsymbol{x}_i \in \mathcal{X}$ is represented by a d-dimensional feature vector $(x_{i1}, x_{i2}, \ldots, x_{id})^\top$ and $S_i \subseteq \mathcal{Y}$ is the set of candidate labels associated with $\boldsymbol{x}_i$. Then, the task of partial label learning is to induce a *multi-class* classifier $f : \mathcal{X} \mapsto \mathcal{Y}$ from $\mathcal{D}$. In partial label learning, the ground-truth label t_i of $\boldsymbol{x}_i$ is assumed to reside in its candidate label set S_i, i.e. $t_i \in S_i$.[1]

Evidently, the major difficulty for partial label learning lies in that the ground-truth label of the PL training example is concealed in its candidate label set and thus not directly accessible to the learning algorithm. Therefore, the common strategy to learn from PL examples is *disambiguation*, i.e. differentiating the modeling outputs of *individual* candidate labels so as to recover ground-truth labeling information. To achieve this, one

• *Min-Ling Zhang, Fei Yu, and Cai-Zhi Tang are with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. Email: {zhangml, yuf, 220141515}@seu.edu.cn*

1. In some literatures, partial label learning is also termed as *ambiguous label learning* [10], [24], *soft label learning* [12], or *superset label learning* [27], [28]. Furthermore, there are some studies admitting noisy candidate label set which might violate the partial label assumption $t_i \in S_i$ [11].

way is *disambiguation by identification* which differentiates individual candidate labels' modeling outputs in a competitive manner. Specifically, the ground-truth label is regarded as a latent variable which is identified via iterative refining procedure such as EM [22], [26], [27], [29], [32]. Another way is *disambiguation by averaging* which differentiates individual candidate labels' modeling outputs in a collaborative manner. Specifically, each candidate label is treated equally where the final prediction is made by averaging their modeling outputs [14], [24].

Although disambiguation serves as an intuitive and practical strategy to learn from PL examples, its effectiveness is largely affected by the *false positive* labels which co-occur with the ground-truth label within candidate label set (i.e. $S_i \setminus \{t_i\}$). For disambiguation by ground-truth label identification, the identified label $\hat{t}_i$ refining in each iteration might turn out to be the false positive label rather than the ground-truth one. Similarly, for disambiguation by candidate label averaging, the essential modeling output yielded by ground-truth label might be overwhelmed by the modeling outputs yielded by false positive labels. Furthermore, for either of the two disambiguation ways, the negative influence introduced by false positive labels would be more pronounced as the size of candidate label set increases.

In view of this, a new strategy to learn from PL examples is proposed in this paper which does not conduct disambiguation by differentiating *individual* candidate labels' modeling outputs. Specifically, by adapting the well-known error-correcting output codes (ECOC) techniques [17], [40], a simple yet effective approach named PL-ECOC is proposed which refrains from conducting disambiguation by making use of the candidate label set as *an entirety*. The key adaptation lies in how the binary classifiers are trained according to the ECOC coding matrix. For each column of the binary coding matrix, one binary classifier is built by utilizing binary training examples derived from the PL training set. Here, any PL training example will be used to derive a positive or negative training example only if its candidate label set *entirely* falls into the positive or negative dichotomy specified by the column coding. During testing phase, class label of the unseen instance is determined via loss-based decoding which takes into account binary classifiers' empirical performance and predictive margin.

To thoroughly evaluate the effectiveness of the proposed disambiguation-free strategy, extensive experiments over a broad range of controlled UCI data sets as well as real-world PL data sets are conducted. Experimental results clearly validate the superior performance of PL-ECOC against several well-established partial label learning approaches.

The rest of this paper is organized as follows. Section 2 briefly reviews related works on partial label learning. Section 3 presents technical details of the proposed PL-ECOC approach. Section 4 reports comparative experimental results against state-of-the-art partial label learning approaches. Finally, Section 5 concludes and indicates several issues for future work.

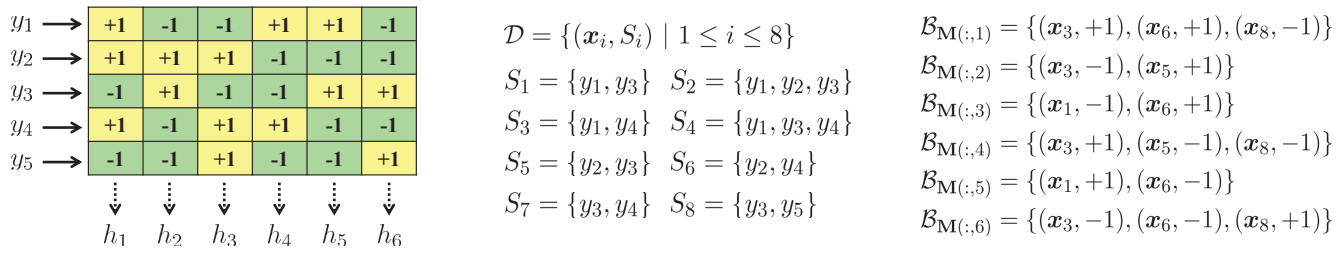
2 Related Work

As shown in Section 1, supervision information conveyed by PL training examples is implicit as the ground-truth label is hidden within the candidate label set. Therefore, partial label learning can be regarded as a *weakly-supervised* learning framework with implicit labeling information. It lies between the two ends of the supervision spectrum, i.e. supervised learning with explicit supervision and unsupervised learning with blind supervision. Partial label learning is related to other popular weakly-supervised learning frameworks such as *semi-supervised learning*, *multi-instance learning* and *multi-label learning*. Nevertheless, the type of weak supervision information handled by partial label learning is different to those counterpart frameworks.

In *semi-supervised learning*, the task is to learn from few labeled examples along with abundant unlabeled examples [9], [41]. For either unlabeled example or PL example, the ground-truth labeling information is not accessible to the learning system. Nonetheless, for unlabeled example the ground-truth label assumes the whole label space while for PL example the ground-truth label is confined within candidate label set. In *multi-instance learning*, the task is to learn from labeled examples each represented by a bag of instances [3], [18]. For either multi-instance example or PL example, the actual correspondence between individual instances and labels is ambiguous. Nonetheless, for multi-instance example the ambiguity arises in the bag of instances while for PL example the ambiguity arises in the set of candidate labels. In *multi-label learning*, the task is to learn from examples each associated with multiple class labels [21], [39]. For either multi-label example and PL example, the labeling information available for the instance is non-unique. Nonetheless, for multi-label example the associated labels are all valid ones while for PL example the associated labels are only candidate ones.

Existing partial label learning approaches aim to fulfill the learning task by disambiguating the candidate label set, which can be achieved in two basic ways. Generally, let $F(\boldsymbol{x}, y; \boldsymbol{\Theta})$ be the modeling output of instance $\boldsymbol{x}$ on label $y \in \mathcal{Y}$. One way towards disambiguation is to treat the ground-truth label as latent variable which is identified as: $\hat{t}_i = \arg\max_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\Theta})$. Here, the model parameters $\boldsymbol{\Theta}$ are iteratively refined by optimizing specific criterion over PL training examples, such as the maximum likelihood criterion: $\sum_{i=1}^{m} \log \left(\sum_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\Theta}) \right)$ [10], [22], [26], [27], [32], or the maximum margin criterion: $\sum_{i=1}^{m} \left(\max_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\Theta}) - \max_{y \notin S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\Theta}) \right)$ [29], [34].

Another way towards disambiguation is to assume equal contribution of each candidate label in the modeling



(a) coding matrix $\mathbf{M}$ (b) PL training set $\mathcal{D}$ (c) binary training set $\mathcal{B}$

Fig. 2. An illustrative example of PL-ECOC's encoding phase. (a) An exemplar 5×6 coding matrix $\mathbf{M}$ with each row corresponding to one class label and each column leading to one binary classifier; (b) An exemplar PL training set with eight PL training examples; (c) The derived binary training set w.r.t. each column coding.

process. For discriminative models, the averaged output over all candidate labels, i.e. $\frac{1}{|S_i|}\sum_{y \in S_i} F(\boldsymbol{x}_i, y; \boldsymbol{\Theta})$, is distinguished from outputs over non-candidate labels, i.e. $F(\boldsymbol{x}_i, y; \boldsymbol{\Theta})$ $(y \notin S_i)$ [14], [31]. For instance-based models, class label for unseen instance $\boldsymbol{x}^*$ is determined by voting among the candidate labels of its neighboring examples indexed in $\mathcal{N}(\boldsymbol{x}^*)$: $f(\boldsymbol{x}^*) = \arg\max_{y \in \mathcal{Y}} \sum_{j \in \mathcal{N}(\boldsymbol{x}^*)} \mathbb{I}(y \in S_j)$ [24], [37].

For either way of disambiguation, its effectiveness would be largely affected due to the existence of false positive labels, i.e. the need to calculate modeling output $F(\boldsymbol{x}_i, y; \boldsymbol{\Theta})$ $(y \in S_i \setminus \{t_i\})$. In the next section, to circumvent potential issues encountered during disambiguation, a simple yet effective disambiguation-free partial label learning approach is proposed.

3 The PL-ECOC Approach

3.1 Binary Decomposition to Multi-Class Classifier

As discussed in Section 1, the ultimate goal of partial label learning is to induce a *multi-class* classifier $f : \mathcal{X} \mapsto \mathcal{Y}$ mapping from the instance space to the label space. For multi-class classifier induction, the arguably most popular mechanism is to transform the learning task into a number of binary learning problems via *one-vs-rest* or *one-vs-one* decomposition.

For one-vs-rest decomposition a total of q binary classifiers are induced, one for each class label y_j $(1 \leq j \leq q)$. Here, each binary classifier is built by treating training examples from y_j as positive ones and the remaining training examples as negative ones. By taking outputs of binary classifiers as predictive confidence on class labels, prediction on unseen instance is determined by choosing the class label with largest classifier output. For one-vs-one decomposition a total of $\binom{q}{2}$ binary classifiers are induced, one for each pair of class labels (y_j, y_k) $(1 \leq j < k \leq q)$. Here, each binary classifier is built by treating training examples from y_j as positive ones and those from y_k as negative ones. By taking outputs of binary classifiers as votes on class labels, prediction on unseen instance is determined by choosing the class label receiving maximal votes from all the binary classifiers.

Unfortunately, under partial label learning scenario, neither the one-vs-rest nor the one-vs-one decomposition mechanism can be employed to induce the multi-class classifier. As the ground-truth label of the PL training example is not directly accessible, training examples needed to build the decomposed binary classifiers can not be properly derived from the PL training set. In the following, a new approach named PL-ECOC is proposed by adapting the ECOC techniques, which is capable of learning from PL training examples by maintaining the simplicity merit of binary decomposition mechanism.

3.2 Partial Label Learning with ECOC

As a well-established mechanism towards multi-class classifier induction, ECOC [17], [40] conducts binary decomposition based on a coding-decoding procedure. In the coding phase, a $q \times L$ coding matrix $\mathbf{M} \in \{+1, -1\}^{q \times L}$ with binary elements is utilized to facilitate the learning process. Here, each row of the coding matrix $\mathbf{M}(j, :)$ corresponds to an L-bits *codeword* for the class label y_j $(1 \leq j \leq q)$. On the other hand, each column of the coding matrix $\mathbf{M}(:, l)$ specifies a *dichotomy* over the label space $\mathcal{Y}$ with positive half $\mathcal{Y}_l^+ = \{y_j \mid \mathbf{M}(j, l) = +1,\ 1 \leq j \leq q\}$ and negative half $\mathcal{Y}_l^- = \{y_j \mid \mathbf{M}(j, l) = -1,\ 1 \leq j \leq q\}$. Accordingly, one binary classifier $h_l : \mathcal{X} \mapsto \mathbb{R}$ is built w.r.t. each column by treating training examples from $\mathcal{Y}_l^+$ as positive ones and those from $\mathcal{Y}_l^-$ as negative ones.

In the decoding phase, given the unseen instance $\boldsymbol{x}^*$, an L-bits codeword is generated by concatenating the (signed) outputs of the L binary classifiers: $\boldsymbol{h}(\boldsymbol{x}^*) = [\mathrm{sign}(h_1(\boldsymbol{x}^*)), \mathrm{sign}(h_2(\boldsymbol{x}^*)), \ldots, \mathrm{sign}(h_L(\boldsymbol{x}^*))]^\top$. Here, $\mathrm{sign}(z)$ returns $+1$ if $z > 0$ and -1 otherwise. Then, the class label whose codeword is *closest* to $\boldsymbol{h}(\boldsymbol{x}^*)$ is returned as the final prediction on $\boldsymbol{x}^*$:

$$f(\boldsymbol{x}^*) = \arg\min_{y_j\ (1 \leq j \leq q)} \ \mathtt{dist}(\boldsymbol{h}(\boldsymbol{x}^*), \mathbf{M}(j, :)) \qquad (1)$$

Here, the distance function $\mathtt{dist}(\cdot, \cdot)$ can be instantiated in various ways such as Hamming distance [17], Euclidean distance [30], loss-based distance [2], [19], etc.

In this paper, we show that the ECOC techniques can be naturally adapted to deal with partial label data. In the encoding phase, the key adaptation lies in how to build the binary classifier w.r.t. each column coding. Specifically, let $\boldsymbol{v} = [v_1, v_2, \ldots, v_q]^\top \in \{+1, -1\}^q$ denote

TABLE 1
The pseudo-code of PL-ECOC.

Inputs:
$\mathcal{D}$: partial label training set $\{(\boldsymbol{x}_i, S_i) \mid 1 \leq i \leq m\}$ ($\boldsymbol{x}_i \in \mathcal{X}$, $S_i \subseteq \mathcal{Y}$, $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{y_1, y_2, \ldots, y_q\}$)
L: ECOC codeword length
$\mathfrak{L}$: binary learner for classifier induction
τ: minimum admissible binary training set size
$\boldsymbol{x}^*$: unseen instance ($\boldsymbol{x}^* \in \mathcal{X}$)

Outputs:
y^*: predicted class label for $\boldsymbol{x}^*$ ($y^* \in \mathcal{Y}$)

Process:
$l = 0$;
while $l \neq L$ **do**
  Randomly generate a q-bits column coding $\boldsymbol{v} = [v_1, v_2, \ldots, v_q]^\top \in \{+1, -1\}^q$;
  Dichotomize the label space into positive half $\mathcal{Y}_{\boldsymbol{v}}^+$ and negative half $\mathcal{Y}_{\boldsymbol{v}}^-$ according to Eq.(2);
  Derive a binary training set $\mathcal{B}_{\boldsymbol{v}}$ from the PL training set $\mathcal{D}$ according to Eq.(3);
  if $|\mathcal{B}_{\boldsymbol{v}}| \geq \tau$ **then**
    $l = l + 1$;
    Set the l-th column of the coding matrix $\mathbf{M}$ to $\boldsymbol{v}$: $\mathbf{M}(:, l) = \boldsymbol{v}$;
    Induce the binary classifier h_l by invoking $\mathfrak{L}$ on $\mathcal{B}_{\boldsymbol{v}}$: $h_l \leftarrow \mathfrak{L}(\mathcal{B}_{\boldsymbol{v}})$;
  end if
end while
Calculate the $q \times L$ performance matrix $\mathbf{H}$ according to Eq.(4);
Calculate the $q \times L$ weight matrix $\hat{\mathbf{H}}$ according to Eq.(5);
Obtain the predictive confidence $h_l(\boldsymbol{x}^*)$ $(1 \leq l \leq L)$ of each binary classifier on $\boldsymbol{x}^*$;
Return $y^* = f(\boldsymbol{x}^*)$ according to Eq.(6).

the q-bits column coding which dichotomizes the label space into positive half $\mathcal{Y}_{\boldsymbol{v}}^+$ and negative half $\mathcal{Y}_{\boldsymbol{v}}^-$:

$$\begin{aligned} \mathcal{Y}_{\boldsymbol{v}}^+ &= \{y_j \mid v_j = +1,\ 1 \leq j \leq q\}; \\ \mathcal{Y}_{\boldsymbol{v}}^- &= \mathcal{Y} \setminus \mathcal{Y}_{\boldsymbol{v}}^+ \end{aligned} \tag{2}$$

Given any PL training example $(\boldsymbol{x}_i, S_i)$, rather than trying to disambiguate the candidate label set S_i associated with $\boldsymbol{x}_i$, PL-ECOC works by regarding S_i as *an entirety* to help build the binary classifier. Under this perspective, a binary training set $\mathcal{B}_{\boldsymbol{v}}$ can be derived from the original PL training set $\mathcal{D}$, where $\boldsymbol{x}_i$ is used as a positive or negative example only if S_i entirely falls into $\mathcal{Y}_{\boldsymbol{v}}^+$ or $\mathcal{Y}_{\boldsymbol{v}}^-$:

$$\begin{aligned} \mathcal{B}_{\boldsymbol{v}} = &\big\{(\boldsymbol{x}_i, +1) \mid S_i \subseteq \mathcal{Y}_{\boldsymbol{v}}^+,\ 1 \leq i \leq m\big\} \bigcup \\ &\big\{(\boldsymbol{x}_i, -1) \mid S_i \subseteq \mathcal{Y}_{\boldsymbol{v}}^-,\ 1 \leq i \leq m\big\} \end{aligned} \tag{3}$$

As shown in Eq.(3), PL training examples whose candidate label sets fail to fall into either dichotomy won't contribute to generate binary training examples for $\mathcal{B}_{\boldsymbol{v}}$. To avoid non-informative binary training set with few examples, PL-ECOC enforces an eligibility condition on the column coding $\boldsymbol{v}$ by controlling the minimum admissible size of $\mathcal{B}_{\boldsymbol{v}}$. Accordingly, one binary classifier h_l is induced by invoking the binary learner $\mathfrak{L}$ on $\mathcal{B}_{\boldsymbol{v}}$, i.e. $h_l \leftarrow \mathfrak{L}(\mathcal{B}_{\boldsymbol{v}})$.

Figure 2 gives an illustrative example of PL-ECOC's encoding phase. Take the first PL training example $(\boldsymbol{x}_1, S_1)$ with $S_1 = \{y_1, y_3\}$ as an example, $\boldsymbol{x}_1$ will be used as a negative example to induce h_3 as S_1 entirely falls into the negative dichotomy of the third column coding, and will be used as a positive example to induce h_5 as S_1 entirely falls into the positive dichotomy of the fifth column coding. Accordingly, as shown in Figure 2, the binary training set w.r.t. each column coding naturally follows from the corresponding coding matrix and PL training set.

In the decoding phase, the key adaptation lies in how to make prediction for unseen instance based on the induced binary classifiers h_l $(1 \leq l \leq L)$. Among various decoding strategies, PL-ECOC chooses to adapt the *loss-weighted decoding* [19] which generalizes Eq.(1) by making use of empirical performance as well as predictive confidence of the induced binary classifiers. Given the coding matrix $\mathbf{M}$, a $q \times L$ performance matrix $\mathbf{H}$ is defined where each element $\mathbf{H}(j, l)$ records the empirical performance of binary classifier h_l $(1 \leq l \leq L)$ w.r.t. the class label y_j $(1 \leq j \leq q)$:

$$\begin{aligned} \mathbf{H}(j, l) = \frac{1}{|\mathcal{D}_j|} \sum_{(\boldsymbol{x}_i, S_i) \in \mathcal{D}_j} [\![\mathrm{sign}(h_l(\boldsymbol{x}_i)) = \mathbf{M}(j, l)]\!] \\ \text{where}\ \ \mathcal{D}_j = \{(\boldsymbol{x}_i, S_i) \mid y_j \in S_i,\ 1 \leq i \leq m\} \end{aligned} \tag{4}$$

Here, $|\cdot|$ returns the cardinality of a set, and $[\![\pi]\!]$ returns 1 if predicate π holds and 0 otherwise.

As shown in Eq.(4), $\mathcal{D}_j$ consists of PL training examples whose candidate label set contains class label y_j. Therefore, $\mathbf{H}(j, l)$ records the fraction of examples in $\mathcal{D}_j$ whose binary predictions yielded by h_l coincide with the binary coding $\mathbf{M}(j, l)$. For instance, to obtain the empirical performance $\mathbf{H}(2, 4)$ w.r.t. the case shown in

TABLE 2
Characteristics of the experimental data sets.

Controlled UCI Data Sets				Configurations
Data set	# Examples	# Features	# Class Labels	
Ecoli	336	7	8	
Dermatology	364	23	6	(I) $r = 1$, $p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 3]**
Vehicle	846	18	4	
Segment	2,310	18	7	(II) $r = 2$, $p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 4]**
Abalone	4,177	7	29	
Satimage	6,435	36	7	(III) $r = 3$, $p \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 5]**
Usps	9,298	256	10	
Pendigits	10,992	16	10	(IV) $p = 1$, $r = 1$, $\epsilon \in \{0.1, 0.2, \ldots, 0.7\}$ **[Figure 6]**
Letter	20,000	16	26	

Real-World Data Sets					
Data set	# Examples	# Features	# Class Labels	Avg. # CLs	Domain
Lost	1,122	108	16	2.23	*automatic face naming* [14]
MSRCv2	1,758	48	23	3.16	*object classification* [27]
BirdSong	4,998	38	13	2.18	*bird song classification* [7]
Soccer Player	17,472	279	171	2.09	*automatic face naming* [35]
LYN 10	18,313	163	11	2.02	*automatic face naming* [23]
LYN 20	19,027	163	21	2.01	
LYN 50	20,308	163	54	1.97	
LYN 100	21,390	163	101	1.94	
LYN 200	22,991	163	219	1.91	

Figure 2, $\mathcal{D}_2$ will consist of three PL training examples $\{(\boldsymbol{x}_2, S_2), (\boldsymbol{x}_5, S_5), (\boldsymbol{x}_6, S_6)\}$ whose candidate label set contains y_2. Furthermore, suppose h_4 classifies examples in $\mathcal{D}_2$ as follows: $h_4(\boldsymbol{x}_2) = +1$, $h_4(\boldsymbol{x}_5) = -1$ and $h_4(\boldsymbol{x}_6) = -1$, then the empirical performance $\mathbf{H}(2, 4)$ turns out to be 0.67 (i.e. 2/3) with $\mathbf{M}(2, 4) = -1$.

To account for the relative performance of each binary classifier, a weight matrix $\hat{\mathbf{H}}$ can be generated by normalizing each row of $\mathbf{H}$:

$$\hat{\mathbf{H}}(j, l) = \frac{\mathbf{H}(j, l)}{\sum_{l=1}^{L} \mathbf{H}(j, l)} \quad (1 \leq j \leq q,\ 1 \leq l \leq L) \tag{5}$$

Given the unseen instance $\boldsymbol{x}^*$, its class label is predicted by the following *loss-weighted* decoding rule:

$$f(\boldsymbol{x}^*) = \arg\min_{y_j\ (1 \leq j \leq q)} \sum_{l=1}^{L} \hat{\mathbf{H}}(j, l) \exp\left(-h_l(\boldsymbol{x}^*) \cdot \mathbf{M}(j, l)\right) \tag{6}$$

As shown in Eq.(6), the closeness to each codeword is measured by the *weighted exponential loss* between the binary classifier's predictive confidence $h_l(\boldsymbol{x}^*)$ and the codeword bit $\mathbf{M}(j, l)$.[2]

Table 1 summarizes the complete encoding phase (Steps 1 to 11) and decoding phase (Steps 12 to 15) of the proposed PL-ECOC approach.[3] During the encoding phase, one potential column coding $\boldsymbol{v}$ is generated at random (Step 3). Once the eligibility condition is satisfied (Step 6), the potential codeword $\boldsymbol{v}$ will be accepted to instantiate a new column of the coding matrix $\mathbf{M}$ (Steps 7 to 8) and then induce the corresponding binary classifier (Step 9). During the decoding phase, a weight matrix $\hat{\mathbf{H}}$ is calculated based on the empirical performance of the induced binary classifiers (Steps 12 to 13). After that, the class label for unseen instance is predicted based on the loss-weighted decoding rule (Steps 14 to 15).

As shown in Table 1, PL-ECOC is free of any disambiguation operation towards the candidate label set which instead is treated in an integrative manner. PL-ECOC inherits the merits of standard ECOC mechanism for being conceptually simple and amenable to different choices of the binary learner $\mathfrak{L}$. As reported in the next section, the performance of PL-ECOC is highly competitive against state-of-the-art partial label learning approaches.

4 Experiments

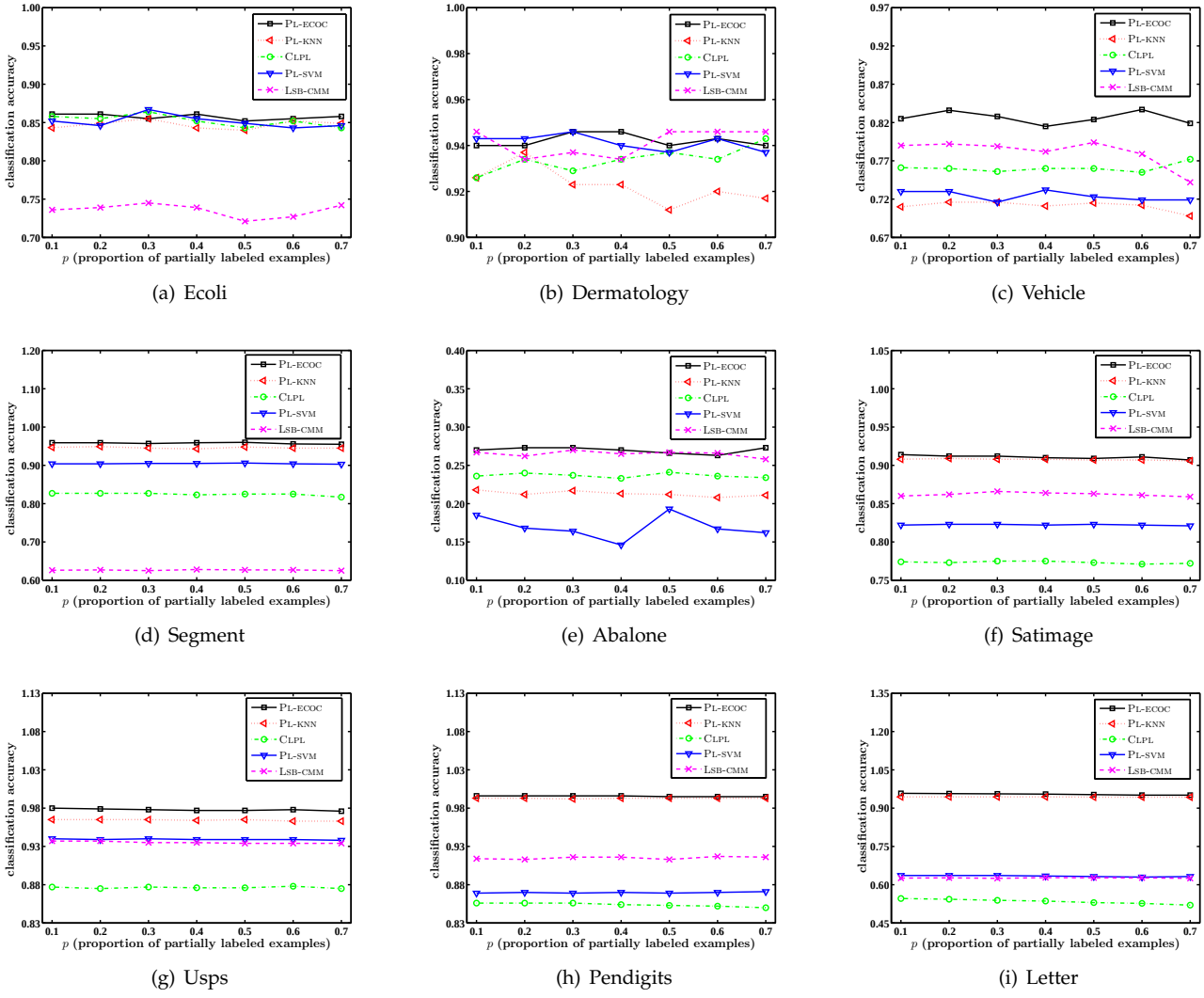
4.1 Experimental Setup

To thoroughly evaluate the performance of PL-ECOC, two series of experiments are conducted on controlled UCI data sets [4] and real-world PL data sets respectively. Characteristics of the experimental data sets are summarized in Table 2.

Following the controlling protocol widely-used in partial label learning studies [10], [14], [27], [34], [37], an *artificial* PL data set can be generated from the 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

2. Compared to the popular decoding rule based on hamming distance: $f(\boldsymbol{x}^*) = \arg\min_{y_j\ (1 \leq j \leq q)} \sum_{l=1}^{L} [\![h_l(\boldsymbol{x}^*) \neq \mathbf{M}(j, l)]\!]$, the loss-weighted decoding rule given in Eq.(6) aims to exploit empirical performance of binary classifiers (i.e. $\mathbf{H}$) to yield better decoding results.

3. Code package for PL-ECOC is publicly-available at: http://cse.seu.edu.cn/PersonalPage/zhangml/Resources.htm#plecoc



(a) Ecoli (b) Dermatology (c) Vehicle

(d) Segment (e) Abalone (f) Satimage

(g) Usps (h) Pendigits (i) Letter

Fig. 3. Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with one false positive candidate label $[r = 1]$).

set of each partially labeled example (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, for each of the nine multi-class UCI data sets, a total of 28 (4×7) controlling parameter configurations have been considered for artificial PL data set generation.

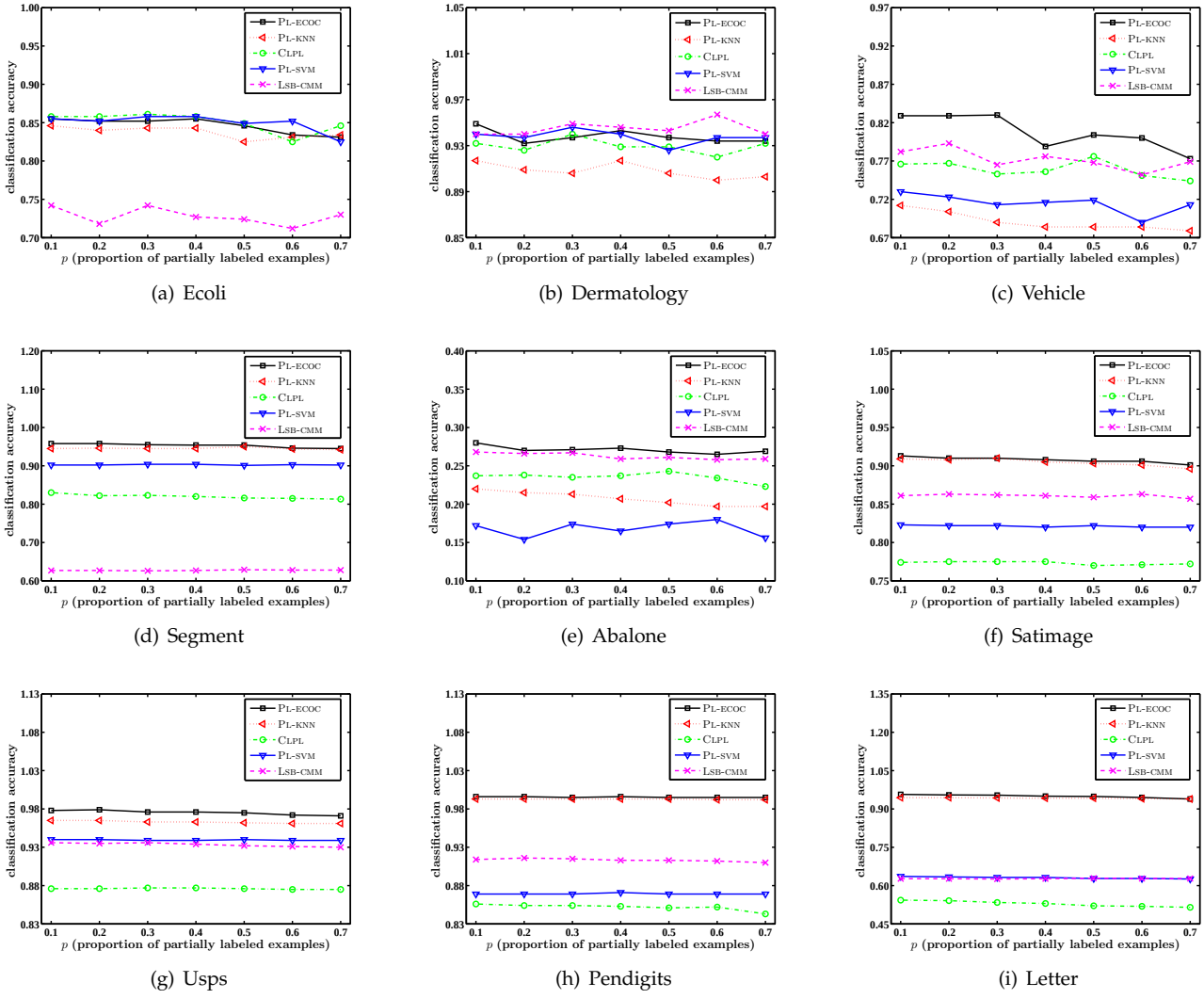
In addition, a number of real-world PL data sets have been collected from several task domains including `Lost` [14], `Soccer Player` [35], `LYN` (Labeled Yahoo! News) [23] from *automatic face naming*, `MSRCv2` [27] from *object classification*, and `BirdSong` [7] from *bird song classification*. For 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 `Num` frequent names from the Labeled Yahoo! News data set [23], five versions of `LYN` data set have been generated (named as `LYN Num`; `Num` $\in \{10, 20, 50, 100, 200\}$).[4] 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 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. # CLs) for each real-world PL data set is also recorded in Table 2.[5]

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

- PL-KNN [24]: A k-nearest neighbor approach to partial label learning which conducts averaging-based

4. In case of ties for the top `Num` frequent names, all the equally frequent names are retained in the `LYN` data set.

5. The real-world PL data sets are publicly available at: http://cse.seu.edu.cn/PersonalPage/zhangml/Resources.htm#partial_data



(a) Ecoli (b) Dermatology (c) Vehicle

(d) Segment (e) Abalone (f) Satimage

(g) Usps (h) Pendigits (i) Letter

Fig. 4. Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with two false positive candidate labels $[r = 2]$).

disambiguation [suggested setup: $k = 10$];

- CLPL [14]: A discriminative approach to partial label learning which conducts averaging-based disambiguation [suggested setup: SVM with squared hinge loss];
- PL-SVM [29]: A maximum margin approach to partial label learning which conducts identification-based disambiguation [suggested setup: regularization parameter pool with $\{10^{-3}, \ldots, 10^{3}\}$];
- LSB-CMM [27]: A maximum likelihood approach to partial label learning which conducts identification-based disambiguation [suggested setup: # mixture components $= q$].

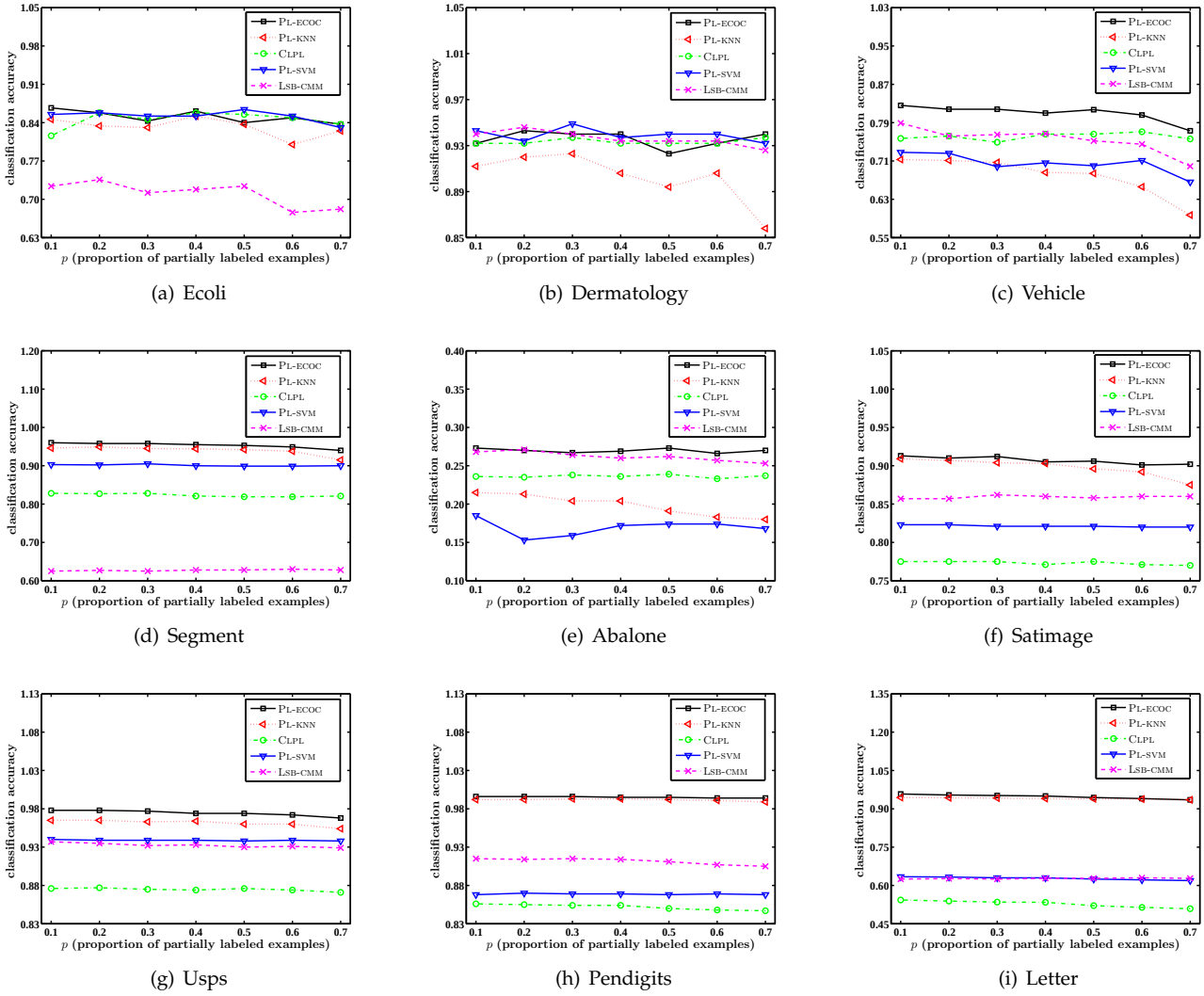
For PL-ECOC, the binary learner $\mathfrak{L}$ is set to be Libsvm [8] and the eligibility parameter τ is set to be one-tenth of the number of PL training examples (i.e. $\frac{1}{10} \cdot |\mathcal{D}|$). Furthermore, the codeword length L for PL-ECOC is set to be $\lceil 10 \cdot \log_2(q) \rceil$ as usually adopted by ECOC-based techniques [2], [30], [40].[6] In this paper, ten-fold cross-validation is performed over each artificial as well as real-world PL data set. Accordingly, the mean predictive accuracy (along with standard deviation) is recorded for all comparing algorithms.

4.2 Experimental Results

4.2.1 *Controlled UCI Data Sets*

Figures 3 to 5 illustrate the mean predictive accuracy of each comparing algorithm as p varies from 0.1 to 0.7 with step-size 0.1 ($r = 1, 2, 3$). Along with the ground-truth label, r additional class labels in $\mathcal{Y}$ will be randomly chosen to instantiate the candidate label set of each partial label example. Figure 6 illustrates the mean predictive accuracy of each comparing algorithm as 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 $y' \neq y$

6. Sensitivity analysis on PL-ECOC's parameters L and τ is reported in Subsection 4.3.1.



(a) Ecoli (b) Dermatology (c) Vehicle

(d) Segment (e) Abalone (f) Satimage

(g) Usps (h) Pendigits (i) Letter

Fig. 5. Classification accuracy of each comparing algorithm changes as p (proportion of partially labeled examples) increases (with three false positive candidate labels $[r = 3]$).

is designated as the coupling label which co-occurs with y in the candidate label set with probability ϵ. Otherwise, any other class label will be randomly chosen to co-occur with y.

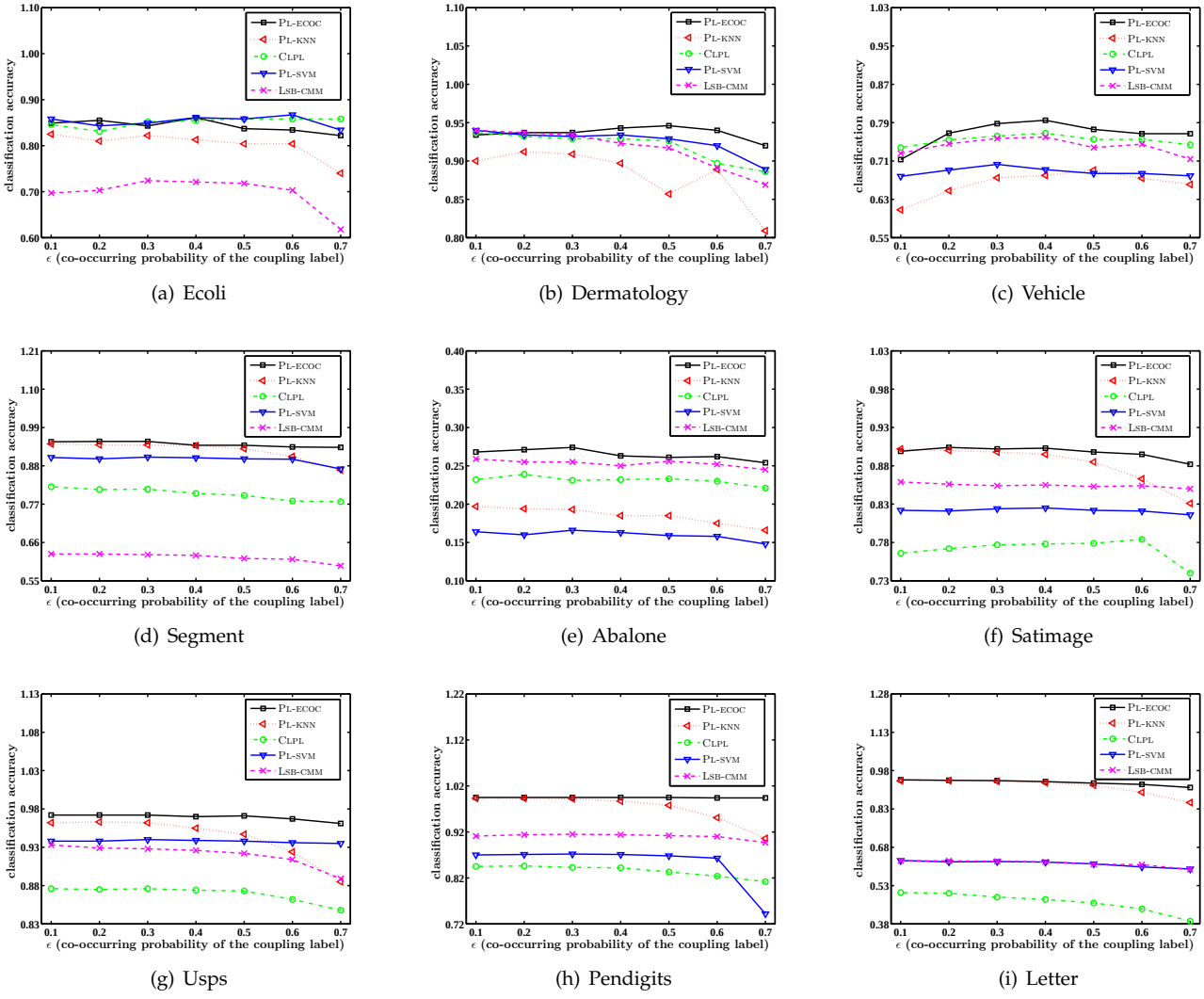
As shown in Figures 3 to 6, the performance of PL-ECOC is highly competitive to the comparing algorithms. Based on Wilcoxon signed ranks test [16], [33] at 0.05 significance level, Table 3 summarizes the win/tie/loss outcomes between PL-ECOC and each comparing algorithm. Here, each statistical test is performed w.r.t. the 7 configurations within each figure for each data set. Based on results of 36 statistical tests (4 figures $\times$ 9 UCI data sets), the following observations can be made:

- Across all the controlling parameter configurations and artificial PL data sets, none of the comparing algorithms have outperformed PL-ECOC significantly;
- Comparing to averaging-based disambiguation approaches (in total), PL-ECOC achieves superior performance against PL-KNN and CLPL in 100% cases (36 out of 36) and 83.3% cases (30 out of 36) respectively;
- Comparing to identification-based disambiguation approaches (in total), PL-ECOC achieves superior performance against PL-SVM and LSB-CMM in 77.8% cases (28 out of 36) and 86.1% cases (31 out of 36) respectively.

4.2.2 Real-World Data Sets

Table 4 reports the performance of each comparing algorithm on the real-world PL data sets, where the outcomes of pairwise t-test at 0.05 significance level are recorded as well. As shown in Table 4, it is impressive to observe that:

- On the `BirdSong`, `LYN 50` and `LYN 200` data sets, the performance of PL-ECOC is superior to all the comparing algorithms;
- On the `Soccer Player`, `LYN 20` and `LYN 100` data sets, the performance of PL-ECOC is comparable



(a) Ecoli (b) Dermatology (c) Vehicle

(d) Segment (e) Abalone (f) Satimage

(g) Usps (h) Pendigits (i) Letter

Fig. 6. 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 [$r = 1$]).

to LSB-CMM and superior to the other comparing algorithms. On the MSRCv2 data set, the performance of PL-ECOC is comparable to PL-SVM and superior to the other comparing algorithms;

- On the Lost data set, the performance of PL-ECOC is inferior to CLPL, comparable to PL-SVM and LSB-CMM, and superior to PL-KNN. On the LYN 10 data set, the performance of PL-ECOC is inferior to LSB-CMM, comparable to PL-SVM, and superior to PL-KNN and CLPL.

4.3 Further Analysis

4.3.1 Parameter Sensitivity

In this subsection, performance sensitivity of the proposed PL-ECOC approach w.r.t. its parameters will be further analyzed.

As shown in Table 1, the codeword length L stands as a crucial parameter for ECOC-based learning approach. Figure 7(a) illustrates how the performance of PL-ECOC changes under varying codeword length ($L = \lceil k \cdot log_2(q) \rceil,\ k \in \{1, 2, \ldots, 15\}$). For the sake of simplicity, MSRCv2, BirdSong and LYN 100 are employed here for illustrative purpose while similar observations can be made on other data sets. As shown in Figure 7(a), the performance of PL-ECOC improves as the codeword length L increases while becomes stable with L approaching $\lceil 10 \cdot log_2(q) \rceil$. Accordingly, the parameter configuration specified for PL-ECOC in Subsection 4.1 corresponds to $L = \lceil 10 \cdot log_2(q) \rceil$.

Furthermore, according to Eq.(3), any PL training example will be excluded from generating the binary training set w.r.t. each column coding if its candidate label set doesn't entirely fall into the positive or negative dichotomy. Let $S^a \subseteq \mathcal{Y}$ denote a candidate label set consisting of a candidate labels, and $\mathcal{Y}_{\boldsymbol{v}}^+$ ($\mathcal{Y}_{\boldsymbol{v}}^-$) denote the positive (negative) dichotomy over $\mathcal{Y}$ w.r.t. to a q-bits column coding $\boldsymbol{v} \in \{+1, -1\}^q$. Suppose that both

TABLE 3
Win/tie/loss outcomes (Wilcoxon signed ranks test at 0.05 significance level) on the classification performance of PL-ECOC against each comparing algorithm on the controlled UCI data sets.

PL-ECOC **against**		Data Sets (names in abbreviation) Eco.	Der.	Veh.	Seg.	Aba.	Sat.	Usp.	Pen.	Let.	**Subtotal**	**In Total**
PL-KNN	[Figure 1]	win	win	win	win	win	win	win	win	win	**9/0/0**	**36/0/0**
	[Figure 2]	win	win	win	win	win	win	win	win	win	**9/0/0**	
	[Figure 3]	win	win	win	win	win	win	win	win	win	**9/0/0**	
	[Figure 4]	win	win	win	win	win	win	win	win	win	**9/0/0**	
CLPL	[Figure 1]	tie	win	win	win	win	win	win	win	win	**8/1/0**	**30/6/0**
	[Figure 2]	tie	win	win	win	win	win	win	win	win	**8/1/0**	
	[Figure 3]	tie	tie	win	win	win	win	win	win	win	**7/2/0**	
	[Figure 4]	tie	win	tie	win	win	win	win	win	win	**7/2/0**	
PL-SVM	[Figure 1]	tie	tie	win	win	win	win	win	win	win	**7/2/0**	**28/8/0**
	[Figure 2]	tie	tie	win	win	win	win	win	win	win	**7/2/0**	
	[Figure 3]	tie	tie	win	win	win	win	win	win	win	**7/2/0**	
	[Figure 4]	tie	tie	win	win	win	win	win	win	win	**7/2/0**	
LSB-CMM	[Figure 1]	win	tie	win	win	tie	win	win	win	win	**7/2/0**	**31/5/0**
	[Figure 2]	win	tie	win	win	win	win	win	win	win	**8/1/0**	
	[Figure 3]	win	tie	win	win	win	win	win	win	win	**8/1/0**	
	[Figure 4]	win	tie	win	win	win	win	win	win	win	**8/1/0**	

TABLE 4
Predictive accuracy (mean±std) of each comparing algorithm on the real-world PL data sets. In addition, •/◦ indicates whether the performance of PL-ECOC is statistically superior/inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significate level).

	PL-ECOC	PL-KNN	CLPL	PL-SVM	LSB-CMM
Lost	0.703±0.052	0.424±0.041•	0.742±0.038◦	0.729±0.040	0.707±0.055
MSRCv2	0.505±0.027	0.448±0.037•	0.413±0.039•	0.482±0.043	0.456±0.031•
BirdSong	0.740±0.016	0.614±0.024•	0.632±0.017•	0.663±0.032•	0.717±0.024•
Soccer Player	0.537±0.020	0.497±0.014•	0.368±0.010•	0.443±0.014•	0.525±0.015
LYN 10	0.694±0.010	0.460±0.012•	0.605±0.013•	0.692±0.009	0.703±0.010◦
LYN 20	0.697±.0.012	0.469±0.015•	0.585±0.010•	0.686±0.011•	0.702±0.011
LYN 50	0.694±0.008	0.472±0.014•	0.540±0.012•	0.666±0.002•	0.679±0.007•
LYN 100	0.680±0.012	0.459±0.010•	0.507±0.011•	0.655±0.010•	0.673±0.010
LYN 200	0.662±0.010	0.457±0.014•	0.462±0.009•	0.636±0.010•	0.648±0.007•

S^a and $\boldsymbol{v}$ are generated with uniform distribution, then the probability of S^a entirely falling into either the positive dichotomy $\mathcal{Y}_{\boldsymbol{v}}^+$ or the negative dichotomy $\mathcal{Y}_{\boldsymbol{v}}^-$ corresponds to:

$$\begin{aligned}&\mathbb{P}\left((S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^+) \vee (S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^-)\right)\\&= 1 - \mathbb{P}\left((S^a \cap \mathcal{Y}_{\boldsymbol{v}}^+ \neq \emptyset) \wedge (S^a \cap \mathcal{Y}_{\boldsymbol{v}}^- \neq \emptyset)\right)\\&= 1 - \frac{1}{2^q \cdot \binom{q}{a}} \sum_{t=0}^{q} \binom{q}{t} \sum_{u=\max(1,a-(q-t))}^{\min(a-1,t)} \binom{t}{u}\binom{q-t}{a-u} \quad (7)\end{aligned}$$

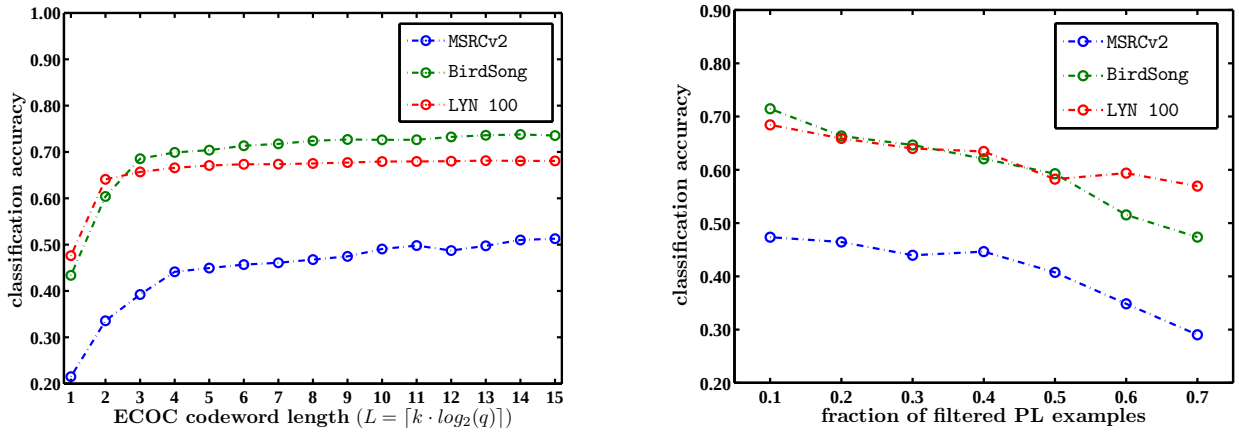
Here, among the $2^q \cdot \binom{q}{a}$ possible joint setups for S^a and $\boldsymbol{v}$, the sum term in Eq.(7) counts the number of joint setups where S^a intersects with both $\mathcal{Y}_{\boldsymbol{v}}^+$ and $\mathcal{Y}_{\boldsymbol{v}}^-$. Based on the combinatorial fact that $\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$, Eq.(7) can be simplified as follows:

$$\begin{aligned}&\mathbb{P}\left((S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^+) \vee (S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^-)\right)\\&= 1 - \frac{1}{2^q} \sum\nolimits_{t=0}^{q} \sum\nolimits_{u=\max(1,a-(q-t))}^{\min(a-1,t)} \binom{a}{u}\binom{q-a}{t-u} \quad (8)\end{aligned}$$

Specifically, the probability $\mathbb{P}\left((S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^+) \vee (S^a \subseteq \mathcal{Y}_{\boldsymbol{v}}^-)\right)$ takes the maximum value of 1 with $a = 1$ and the minimum value of 2^{1-q} with $a = q$.

As shown in Table 1, to account for the possibility of candidate label set not entirely falling into either dichotomy, PL-ECOC employs an eligibility parameter τ to avoid deriving non-informative binary training set with few examples. Table 5 reports the performance of PL-ECOC on the `MSRCv2`, `BirdSong` and `LYN 100` data sets with τ taking fractional size of the training set, i.e. $\tau = \eta \cdot |\mathcal{D}|$ ($\eta \in \{0.1, 0.2, 0.3, 0.4\}$). As shown in Table 5, the performance of PL-ECOC is generally not sensitive to varying value of τ. Accordingly, the other parameter configuration specified for PL-ECOC in Subsection 4.1 corresponds to $\tau = 0.1 \cdot |\mathcal{D}|$.

Once the eligibility condition $|\mathcal{B}_{\boldsymbol{v}}| \geq \tau$ is satisfied (Table 1, Step 6), the size of binary training set would be proportional to the size of original PL training set. To show how PL-ECOC performs under different training set sizes, Table 6 reports the performance of each comparing algorithm on sampled subset of `MSRCv2`, `BirdSong` and `LYN 100` with sampling rate r ($r \in \{30\%, 45\%, 60\%, 75\%, 90\%\}$). As shown in Table 6, PL-



(a) varying codeword length

(b) varying filtering fraction

Fig. 7. Classification accuracy of PL-ECOC changes as: (a) L (codeword length) increases from $\lceil log_2(q) \rceil$ to $\lceil 15 \cdot log_2(q) \rceil$ with step-size $\lceil log_2(q) \rceil$; (b) α (fraction of filtered PL examples with least candidate set size) increases from 0.1 to 0.7 with step-szie 0.1.

TABLE 5
Predictive accuracy (mean±std) of PL-ECOC with the eligibility parameter τ set to be $\eta \cdot |\mathcal{D}|$ ($\eta \in \{0.1, 0.2, 0.3, 0.4\}$).

	MSRCv2	BirdSong	LYN 100
$\eta = 0.1$	0.505±0.027	0.740±0.016	0.680±0.012
$\eta = 0.2$	0.486±0.038	0.723±0.012	0.679±0.010
$\eta = 0.3$	0.479±0.037	0.730±0.020	0.678±0.010
$\eta = 0.4$	0.487±0.035	0.730±0.019	0.674±0.008

ECOC maintains its performance advantage against the comparing algorithms under different sampling rates. Specifically, there is a general trend that the advantage becomes more pronounced as the sampling rate increases (from $r = 45\%$ on).

4.3.2 Algorithmic Properties

In this subsection, several algorithmic properties of the proposed PL-ECOC approach will be further analyzed.

As shown in Figures 3 to 5, it is somewhat surprising that in some cases the performance of comparing algorithms on controlled UCI data sets doesn't decrease as expected when p (proportion of partially labeled examples) increases. Correspondingly, similar observations have also been reported in related literatures [27], [34]. One reason underlying this counter-intuitive trend might lie in the controlling protocol used for UCI data sets, where the $(1 - p) \times 100\%$ training examples with unique labeling play major role in contributing to the generalization ability of induced predictive model. Furthermore, it is worth noting that the controlled UCI data sets are generated *artificially* whose characteristics won't be as representative as real-world PL data sets. Figure 7(b) illustrates how the performance of PL-ECOC changes when $\alpha \times 100\%$ PL examples with least candidate label set size are filtered out from the data set ($\alpha \in \{0.1, 0.2, \ldots, 0.7\}$). It is obvious that as the difficulty of the resulting PL learning task increases (i.e. with larger number of average candidate labels for the filtered data set), the performance of PL-ECOC decreases as expected on the MSRCv2, BirdSong and LYN 100 data sets.

PL-ECOC can be regarded as an ensemble learning approach which leverages a number of base binary classifiers induced w.r.t. each column coding to yield the predictive model. Interestingly, one can also apply ensemble learning techniques to PL-ECOC by treating the proposed approach as the base learner. Specifically, Table 7 reports the performance of PL-ECOC ensemble based on the popular *bagging* techniques [5], [40], where the predictive model is built by ensembling N PL-ECOC classifiers each induced from one bootstrapped sampling of original training set ($N \in \{5, 10, 15, 20, 25\}$). As shown in Table 7, on the MSRCv2, BirdSong and LYN 100 data sets, the performance of PL-ECOC ensemble doesn't have significant changes with varying ensemble sizes, which is also comparable to the plain PL-ECOC as shown in Table 4. These results indicate that PL-ECOC serves as a *stable* learner [6], [40] whose learning procedure is less sensitive to the perturbation over training set.

Table 8 also reports the running time (in seconds) of PL-ECOC as well as the comparing algorithms, measured within Matlab environment equipped with Intel i7-6700 CPU. As shown in Table 8, the computational complexity of PL-ECOC is close or less than most comparing algorithms (CLPL, PL-SVM, LSB-CMM) on medium-scale data sets MSRCv2 and BirdSong, while significantly increases on large-scale data set LYN 100 due to longer codeword length and larger binary training set size.

4.3.3 Binary Learner

Among the four comparing algorithms given in Subsection 4.1, three of them are tailored towards *concrete* learning techniques. Specifically, PL-KNN, PL-SVM and

TABLE 6
Predictive accuracy (mean±std) of each comparing algorithm on sampled subset of MSRCv2, BirdSong and LYN 100 (sampling rate $r \in \{30\%, 45\%, 60\%, 75\%, 90\%\}$). In addition, •/◦ indicates whether the performance of PL-ECOC is statistically superior/inferior to the comparing algorithm on each sampled data set (pairwise t-test at 0.05 significate level).

		PL-ECOC	PL-KNN	CLPL	PL-SVM	LSB-CMM	win/tie/loss counts
$r = 30\%$	MSRCv2	0.426±0.095	0.433±0.054	0.374±0.046•	0.445±0.064	0.320±0.051•	**9/3/0**
	BirdSong	0.652±0.049	0.536±0.035•	0.589±0.058•	0.621±0.054	0.585±0.043•	
	LYN 100	0.566±0.023	0.413±0.016•	0.469±0.026•	0.548±0.028•	0.556±0.020•	
$r = 45\%$	MSRCv2	0.422±0.046	0.414±0.046	0.382±0.043•	0.438±0.040	0.390±0.036•	**7/4/1**
	BirdSong	0.692±0.027	0.583±0.029•	0.618±0.029•	0.641±0.037•	0.662±0.023	
	LYN 100	0.593±0.025	0.433±0.015•	0.484±0.012•	0.612±0.012◦	0.598±0.016	
$r = 60\%$	MSRCv2	0.455±0.051	0.423±0.033	0.410±0.044•	0.448±0.038	0.390±0.042•	**8/4/0**
	BirdSong	0.701±0.032	0.594±0.039•	0.627±0.033•	0.660±0.034•	0.633±0.031•	
	LYN 100	0.634±0.015	0.442±0.012•	0.503±0.011•	0.635±0.009	0.636±0.012	
$r = 75\%$	MSRCv2	0.460±0.045	0.439±0.038	0.386±0.044•	0.451±0.045	0.401±0.032•	**8/4/0**
	BirdSong	0.717±0.022	0.613±0.028•	0.629±0.026•	0.657±0.020•	0.643±0.023•	
	LYN 100	0.663±0.013	0.455±0.011•	0.508±0.014•	0.645±0.014	0.647±0.012	
$r = 90\%$	MSRCv2	0.500±0.039	0.440±0.036•	0.416±0.048•	0.466±0.032	0.430±0.030•	**10/2/0**
	BirdSong	0.741±0.018	0.615±0.019•	0.634±0.024•	0.664±0.021•	0.632±0.030•	
	LYN 100	0.675±0.012	0.459±0.017•	0.507±0.012•	0.650±0.010•	0.668±0.010	

TABLE 7
Predictive accuracy (mean±std) of an ensemble of PL-ECOC with N times of bootstrapped samplings ($N \in \{5, 10, 15, 20, 25\}$).

	MSRCv2	BirdSong	LYN 100
$N = 5$	0.488±0.032	0.737±0.013	0.673±0.011
$N = 10$	0.495±0.032	0.740±0.013	0.678±0.009
$N = 15$	0.497±0.023	0.743±0.013	0.679±0.010
$N = 20$	0.504±0.024	0.744±0.010	0.677±0.009
$N = 25$	0.502±0.031	0.746±0.008	0.678±0.008

TABLE 8
Running time (in seconds) of the comparing algorithms on the MSRCv2, BirdSong and LYN 100 data sets.

	MSRCv2	BirdSong	LYN 100
PL-ECOC	19.153	86.560	78610.266
PL-KNN	0.609	1.643	103.035
CLPL	20.449	143.355	7136.593
PL-SVM	37.051	96.387	875.185
LSB-CMM	1078.023	1648.418	12786.078

LSB-CMM are adapted from *k-nearest neighbor* [1], *support vector machines* [15], and *nonparametric Bayesian* [20] respectively to fit the partial label training examples. On the other hand, CLPL works in similar way as PL-ECOC by transforming the original partial label learning problem into binary learning problem such that any binary learner can be applied thereafter. Specifically, CLPL transforms each PL training example $(\boldsymbol{x}_i, S_i) \in \mathcal{D}$ into one *positive* example by aggregating all candidate labels, and $q - |S_i|$ *negative* examples each for one non-candidate label.

Considering that both PL-ECOC and CLPL rely on the choice of binary learner $\mathfrak{L}$ to instantiate the learning algorithms, Table 9 reports the performance of PL-ECOC and CLPL on the MSRCv2, BirdSong, and LYN 100 data sets instantiated with different choices of base learner $\mathfrak{L}$ ($\mathfrak{L} \in$ {SVM, Logistic Regression (LR), Perceptron (PT)}). As shown in Table 9, the choice of base learner does have significant influence on the performance of both algorithms. Furthermore, PL-ECOC performs significantly better than CLPL with base learners *SVM* and *Perceptron* while is inferior to CLPL with base learner *Logistic Regression*.

TABLE 9
Predictive accuracy (mean±std) of PL-ECOC and CLPL instantiated with different binary learner $\mathfrak{L}$ ($\mathfrak{L} \in$ {SVM, Logistic Regression (LR), Perceptron (PT)}). In addition, •/◦ indicates whether the performance of PL-ECOC is statistically superior/inferior to CLPL on each data set (pairwise t-test at 0.05 significate level)

		MSRCv2	BirdSong	LYN 100
$\mathfrak{L}$=SVM	PL-ECOC	0.505±0.027	0.740±0.016	0.680±0.012
	CLPL	0.413±0.039•	0.632±0.017•	0.507±0.011•
$\mathfrak{L}$=LR	PL-ECOC	0.312±0.051	0.598±0.044	0.468±0.025
	CLPL	0.408±0.039◦	0.646±0.023◦	0.579±0.012◦
$\mathfrak{L}$=PT	PL-ECOC	0.458±0.045	0.711±0.025	0.463±0.012
	CLPL	0.334±0.030•	0.331±0.045•	0.282±0.008•

Summary: As analyzed in Subsections 4.3.1 to 4.3.3, the following observations on the effectiveness of PL-ECOC can be made: (a) PL-ECOC is not sensitive w.r.t. to the codeword length parameter L (Figure 7(a)) and the eligibility parameter τ (Table 5); (b) The performance advantage of PL-ECOC against the comparing algorithms is more pronounced with larger training set size (Table 6); (c) PL-ECOC is a stable learner whose performance is robust to the perturbation over training set (Table 7); (d) The choice of base learner has significant effects on the performance of PL-ECOC (Table 9).

5 Conclusion

Existing partial label learning approaches aim to learn from PL training examples by trying to disambiguate their candidate label sets. In this paper, an extension to our preliminary research [36] is presented which learns from PL training examples in a disambiguation-free manner. Specifically, by adapting the popular E-COC techniques, a novel partial label learning approach named PL-ECOC is proposed by making use of the candidate label set as an entirety. Comprehensive experimental studies show that disambiguation-free strategy is a promising direction for learning from partial label data.

For each column coding, there are some PL training examples which will be excluded from generating the corresponding binary training set (Eq.(3)). Therefore, it is interesting to investigate effective ways to make full use of those excluded PL training examples. Furthermore, coding strategies other than the random one need to be investigated when the partial label sets of PL training examples exhibit certain structures (e.g. ordinal structure over class labels [34]). In the future, it is also important to explore other techniques to enable disambiguation-free partial label learning.

References

[1] D. W. Aha, "Special issue editorial on lazy learning," *Artificial Intelligence Review*, vol. 11, pp. 7–10, 1997.

[2] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *Journal of Machine Learning Research*, vol. 1, no. Dec, pp. 113–141, 2000.

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

[4] K. Bache and M. Lichman, "UCI machine learning repository. School of Information and Computer Sciences, University of California, Irvine," 2013. [Online]. Available: [http://archive.ics.uci.edu/ml]

[5] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[6] ——, "Heuristics of instability and stabilization in model selection," *Annals of Statistics*, vol. 24, no. 6, pp. 2350–2383, 1996.

[7] 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*, Beijing, China, 2012, pp. 534–542.

[8] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, p. Article 27, 2011.

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

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

[11] J. Cid-Sueiro, "Proper losses for learning from partial labels," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Weinberger, Eds. Cambridge, MA: MIT Press, 2012, pp. 1574–1582.

[12] E. Côme, L. Oukhellou, T. Denoeux, and P. Aknin, "Mixture model estimation with soft labels," in *Advances in Soft Computing 48*, D. Dubois, M. A. Lubiano, H. Prade, M. A. Gil, P. Grzegorzewski, and O. Hryniewicz, Eds. Berlin: Springer, 2008, pp. 165–174.

[13] 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*, Miami, FL, 2009, pp. 919–926.

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

[15] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. New York, NY: Cambridge University Press, 2000.

[16] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.

[17] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problem via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, no. 1, pp. 263–286, 1995.

[18] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple-instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 1-2, pp. 31–71, 1997.

[19] S. Escalera, O. Pujol, and P. Radeva, "On the decoding process in ternary error-correcting output codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 120–134, 2010.

[20] S. J. Gershman and D. M. Blei, "A tutorial on Bayesian nonparametric models," *Journal of Mathematical Psychology*, vol. 56, no. 1, pp. 1–12, 2012.

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

[22] Y. Grandvalet and Y. Bengio, "Learning from partial labels with minimum entropy," Center for Interuniversity Research and Analysis of Organizations, Québec, Canada, Tech. Rep., 2004.

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

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

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

[26] R. Jin and Z. Ghahramani, "Learning with multiple labels," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds. Cambridge, MA: MIT Press, 2003, pp. 921–928.

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

[28] ——, "Learnability of the superset label learning problem," in *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014, pp. 1629–1637.

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

[30] O. Pujol, S. Escalera, and P. Radeva, "An incremental node embedding technique for error correcting output codes," *Pattern Recognition*, vol. 41, no. 2, pp. 713–725, 2008.

[31] C.-Z. Tang and M.-L. Zhang, "Confidence-rated discriminative partial label learning," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, San Francisco, CA, 2017, pp. 2611–2617.

[32] P. Vannoorenberghe and P. Smets, "Partially supervised learning by a credal EM approach," in *Lecture Notes in Computer Science 3571*, L. Godo, Ed. Berlin: Springer, 2005, pp. 956–967.

[33] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, pp. 80–83, 1945.

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

[35] 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*, Portland, OR, 2013, pp. 708–715.

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

[37] 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, Buenos Aires, Argentina, 2015, pp. 4048–4054.

[38] 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*, San Francisco, CA, 2016, pp. 1335–1344.

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

[40] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL: Chapman & Hall/CRC, 2012.

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