Network Cooperation with Progressive Disambiguation for Partial Label Learning

Yao Yao[1], Chen Gong[1] (✉), Jiehui Deng[1], and Jian Yang[1,2] (✉)

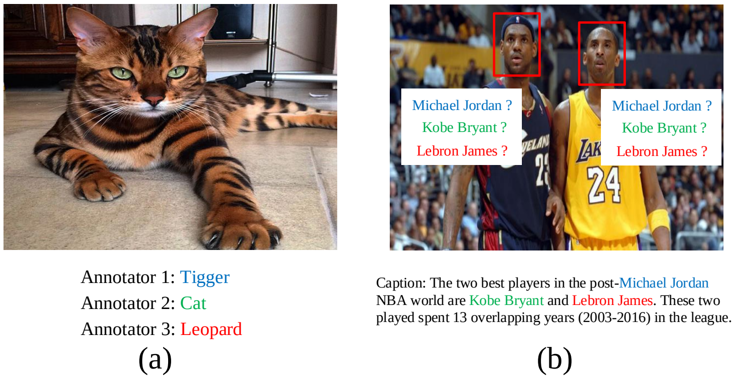
[1] PCA Lab, the Key Laboratory of Intelligent Perception and Systems for
High-Dimensional Information of Ministry of Education, School of Computer Science
and Engineering,
Nanjing University of Science and Technology, China
[2] Jiangsu Key Lab of Image and Video Understanding for Social Security
{yaoyao, chen.gong, jhdeng, csjyang}@njust.edu.cn

Abstract. Partial Label Learning (PLL) aims to train a classifier when each training instance is associated with a set of candidate labels, among which only one is correct but is not accessible during the training phase. The common strategy dealing with such ambiguous labeling information is to disambiguate the candidate label sets. Nonetheless, existing methods ignore the disambiguation difficulty of instances and adopt the single-trend training mechanism. The former would lead to the vulnerability of models to the false positive labels and the latter may arouse error accumulation problem. To remedy these two drawbacks, this paper proposes a novel approach termed "Network Cooperation with Progressive Disambiguation" (NCPD) for PLL. Specifically, we devise a progressive disambiguation strategy of which the disambiguation operations are performed on simple instances firstly and then gradually on more complicated ones. Therefore, the negative impacts brought by the false positive labels of complicated instances can be effectively mitigated as the disambiguation ability of the model has been strengthened via learning from the simple instances. Moreover, by employing artificial neural networks as the backbone, we utilize a network cooperation mechanism which trains two networks collaboratively by letting them interact with each other. As two networks have different disambiguation ability, such interaction is beneficial for both networks to reduce their respective disambiguation errors, and thus is much better than the existing algorithms with single-trend training process. Extensive experimental results on various benchmark and practical datasets demonstrate the superiority of our NCPD approach to other state-of-the-art PLL methods.

Keywords: Weakly-supervised learning · Partial label learning · Progressive disambiguation · Network cooperation.

1 Introduction

Partial Label Learning (PLL), which is also known as *superset label learning* [9, 18, 19] and *ambiguous label learning* [15], is one of the emerging research fields



Annotator 1: Tigger
Annotator 2: Cat
Annotator 3: Leopard

(a)

Caption: The two best players in the post-Michael Jordan NBA world are Kobe Bryant and Lebron James. These two played spent 13 overlapping years (2003-2016) in the league.

(b)

Fig. 1: Two example applications of PLL. (a) In crowdsourcing, some annotators may mistakenly label the picture of a cat with "Tigger" or "Leopard" due to their limited cognitive ability. In this case, the query image contains three labels but only one of them is correct. (b) A newsletter contains an image and the corresponding text caption, from which we can roughly know that Michael Jordan, Kobe Bryant, and Lebron James may in the image. However, we can not figure out the concrete correspondence between the faces and the names.

in weakly-supervised learning [5, 10, 26, 35]. PLL learns from ambiguous labeling information where each training instance is associated with multiple candidate labels and only one of them is valid. Due to the prevalence of ambiguous labeling in real-world scenarios, PLL has many practical applications such as crowdsourcing [9], image classification [4, 6, 21, 29], web mining [22], etc (see Fig. 1).

Formally, let $\mathcal{X} \in \mathbb{R}^d$ denote the d-dimensional input space and $\mathcal{Y} = \{1, 2, \cdots, c\}$ denote the label space with c class labels. The task of PLL is to induce a classifier $f : \mathcal{X} \to \mathcal{Y}$ from the partial label training set $\mathcal{D} = \{(\mathbf{x}_i, \mathcal{S}_i) | 1 \leq i \leq N\}$, where $\mathbf{x}_i \in \mathcal{X}$ is a d-dimensional feature vector and $\mathcal{S}_i \subseteq \mathcal{Y}$ is the corresponding candidate label set of $\mathbf{x}_i$. Particularly, the basic assumption under PLL framework is that the latent groundtruth label y_i of $\mathbf{x}_i$ lies in $\mathcal{S}_i$, *i.e.*, $\mathrm{y}_i \in \mathcal{S}_i$, whereas it is not directly accessible during the training phase.

To learn from such partially labeled instances with ambiguously supervised information, the common strategy is to disambiguate the set of candidate labels of each training instance, namely to detect the unique correct label among multiple candidate labels. There are mainly two classes of methods for such disambiguation operation, namely average-based methods and identification-based methods. Average-based methods treat all candidate labels equally by assuming that they contribute equally to the trained classifier and the prediction is made by averaging their model outputs [15, 36]. These methods share a common deficiency that the effectiveness of the model is greatly affected by the false positive labels in the candidate label sets, which leads to the suppression of groundtruth label by these false positive labels. Identification-based methods address this shortcoming via considering groundtruth label as a latent variable and gradually identifying it by iterative procedures such as Expectation Maximization (EM) [16, 24, 31]. One potential drawback of identification-based methods is that rather than recovering the latent groundtruth labels, the identified labels might turn out to be false positive and they can hardly be rectified in the subsequent iterations.

In a word, existing methods are vulnerable to false positive labels in the candidate label sets. There are two critical reasons that account for this. Firstly, existing approaches scarcely take the disambiguation difficulty of instances into account, and the disambiguation operations are performed on every training instance all at once. In this case, when the instance is complicated and difficult to classify, their models are likely to mistakenly regard the false positive label as the latent groundtruth label, which will mislead the training process and ultimately impair the disambiguation ability of the models. Secondly, the training process of existing methods are all single-trend, which indicates that the data disambiguated at the current step will be directly transferred back to the model itself in the following steps. Under this circumstance, once the identified labels turn out to be false positive, they would be difficult to correct in the succeeding iterations and thereby raising the error accumulation problem, which will severely degrade their performances.

To address these two shortcomings, this paper proposes a novel approach which employs a progressive disambiguation strategy combined with a network cooperation mechanism for PLL, which is termed "Network Cooperation with Progressive Disambiguation" ("NCPD" for short). Specifically, to address the problem of ignoring the disambiguation difficulty of instances, we devise a progressive disambiguation strategy which disambiguates simple instances firstly and then gradually disambiguates more complicated ones. Through learning from the simple instances, the disambiguation ability of the model can be improved steadily. With the proceeding of training process, the model is capable of disambiguating the complicated instances precisely. As a consequence, the negative impacts brought by the false positive labels, especially those of complicated instances, can be effectively mitigated. To settle the error accumulation problem caused by the single-trend training mechanism of traditional methods, we employ Artificial Neural Networks (ANNs) [14] as the backbone and utilize a network cooperation mechanism which trains two networks collaboratively by letting them interact with each other. That is to say, two networks disambiguate the training instances independently in the forward propagation phase and then back propagate the data disambiguated by its peer network. As two networks have different ability and can disambiguate training instances at different levels, such interaction is beneficial for both networks to learn from each other and thus their respective disambiguation errors can be reduced. As a result, the error accumulation problem can be significantly alleviated, and that is why we adopt such network cooperation mechanism rather than the existing single-trend training process. Intensive experiments on multiple datasets substantiate the superiority of our proposed NCPD approach to the state-of-the-art methodologies.

The rest of this paper is organized as follows. We review the related works in Section 2, and introduce the proposed NCPD approach in Section 3. Section 4 reports the experimental results, followed by the conclusion in Section 5.

2 Related Work

Existing algorithms dealing with partially labeled instances can be roughly grouped into the following two classes, *i.e.*, average-based methods and identification-based methods.

The average-based methods treat all candidate labels equally and the prediction is made by averaging their model outputs. For example, the work [15] straightforwardly generalizes the k-nearest neighbor classifier to resolve the PLL problem by predicting the label of a test instance $\mathbf{x}$ via the voting strategy among the candidate labels of its neighbors. That is to say, $f(\mathbf{x}) = \text{argmax}_{y \in \mathcal{Y}} \sum_{i \in \mathcal{N}_{(\mathbf{x})}} \mathbb{I}(y \in \mathcal{S}_i)$, where $\mathcal{N}(\mathbf{x})$ denotes the neighbors of the test instance $\mathbf{x}$ and $\mathbb{I}(\cdot)$ is the indicator function. Zhang *et al.* [36] also propose a model of which the predictions of unseen instances are made by the weighted averaging over the candidate labels of their neighbors. Cour *et al.* [6] propose a convex learning method and decide the groundtruth label by averaging the outputs from all candidate labels, *i.e.*, $\frac{1}{|\mathcal{S}_i|} \sum_{y \in \mathcal{S}_i} F(\mathbf{x}, y; \Theta)$ with Θ being the model parameters. Average-based methods are intuitive and are easy to implement. However, these methods share a critical shortcoming that the outputs from false positive labels may overwhelm the groundtruth labels' outputs, which will severely degrade their performances.

The identification-based methods regard the unique groundtruth label as a latent variable and identify it as $\text{argmax}_{y \in \mathcal{S}_i} F(\mathbf{x}, y; \Theta)$. Maximum likelihood criterion and maximum margin criterion are the two most widely-used learning strategies to identify groundtruth labels. Based on EM procedure, the methods [16, 19] train their models by optimizing the maximum likelihood function $\sum_{i=1}^{n} \log(\sum_{y \in \mathcal{S}_i} F(\mathbf{x}, y; \Theta))$. The work [24] maximizes the margin between outputs from candidate labels and that from non-candidate labels to refine groundtruth labels, and the corresponding objective function is $\sum_{i=1}^{n} (\max_{y \in \mathcal{S}_i} F(\mathbf{x}, y; \Theta) - \max_{y \notin \mathcal{S}_i} F(\mathbf{x}, y; \Theta))$. Nonetheless, the above margin ignores the predictive difference between the latent groundtruth label and other candidate labels. To address this problem, Yu *et al.* [31] maximize the margin between the groundtruth label and other labels, *i.e.*, $\sum_{i=1}^{n} (F(\mathbf{x}_i, \mathrm{y}_i; \Theta) - \max_{y \neq \mathrm{y}_i} F(\mathbf{x}_i, y; \Theta))$ where y_i denotes the groundtruth label of $\mathbf{x}_i$. Moreover, by applying the idea of self-paced learning, Lyu *et al.* [23] propose a novel algorithm which utilizes the maximum margin criterion to detect the groundtruth label. Differently, Feng *et al.* [8] balance the minimum approximation loss and the maximum infinity norm of the outputs to differentiate the unique groundtruth label from false positive labels. Chen *et al.* [4] eliminate a proportion of the least likely candidates in each iteration to enhance the discriminability of their proposed approach. One potential shortcoming of identification-based methods is that the identified label in the current iteration may turn out to be false positive and they can hardly be rectified in the subsequent iterations.

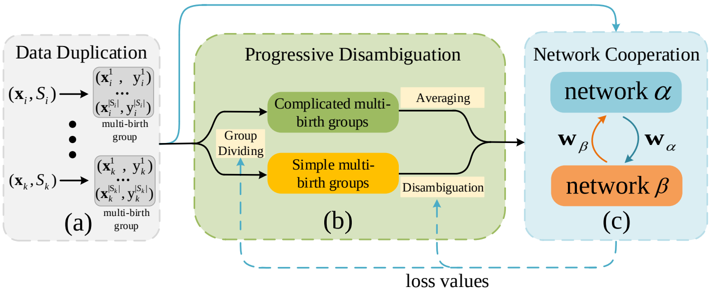


Fig. 2: The framework of our method. (a) indicates the data duplication scheme which transforms each partially labeled instance into a multi-birth group. After that, we feed the transformed data into the networks and thus their corresponding loss values can be obtained (the blue line). (b) presents the process of dividing multi-birth groups into two levels of difficulty and then calculating the confidence scores of instances among them according to the incurred loss values. (c) denotes the network cooperation mechanism where two networks interact with each other via exchanging their respective confidence scores of instances (*i.e.*, $\mathbf{w}_\alpha$ and $\mathbf{w}_\beta$) for back propagation.

In a word, although the aforementioned methods have achieved good performances to some degree, they still suffer from two severe drawbacks, *i.e.*, ignoring the disambiguation difficulty of instances and adopting the unreliable single-trend training process, and both of them will degrade their performances as mentioned in the introduction. Therefore, this paper presents a novel algorithm termed NCPD which will be introduced in the next section.

3 The Proposed NCPD Approach

In this section, we introduce the NCPD approach of which the architecture is illustrated in Fig. 2. To facilitate the disambiguation process, we firstly employ a data duplication scheme which transforms each partially labeled instance into a multi-birth group[3] (Fig. 2 (a)). Afterwards, by dividing these multi-birth groups into two levels of difficulty (*i.e.*, “simple” and “complicated”), we can calculate the confidence scores of instances among them via averaging or disambiguation (Fig. 2 (b)). Finally, two networks collaborate with each other through exchanging the confidence scores of instances generated by them independently to compute their respective back propagated loss (Fig. 2 (c)). We will detail these critical steps in the following sections.

3.1 Data Duplication

We denote $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ as the training set with each column $\mathbf{x}_i$ ($i = 1, 2, \ldots, N$) representing the feature vector of the i-th instance and N denotes

[3] The notion of “multi-birth group” will be detailed later in Section 3.1.

the total number of training instances. Besides, we represent the candidate label set of $\mathbf{x}_i$ as $\mathcal{S}_i = \{\mathrm{y}_i^1, \mathrm{y}_i^2, \ldots, \mathrm{y}_i^{|\mathcal{S}_i|}\}$, where $|\mathcal{S}_i|$ denotes the cardinality of $\mathcal{S}_i$.

To pave the way for the subsequent disambiguation operations, we adopt a data duplication scheme on the original partially labeled training dataset. Specifically, for an arbitrary training instance $\mathbf{x}_i$ and its corresponding candidate label set $\mathcal{S}_i$, we first duplicate $\mathbf{x}_i$ into $|\mathcal{S}_i|$ replicas, *i.e.*, $\mathbf{x}_i^1$, $\mathbf{x}_i^2$, ..., and $\mathbf{x}_i^{|\mathcal{S}_i|}$, and each replica is identical to the original feature vector $\mathbf{x}_i$. After that, we decompose the corresponding candidate label set $\mathcal{S}_i = \{\mathrm{y}_i^1, \mathrm{y}_i^2, \ldots, \mathrm{y}_i^{|\mathcal{S}_i|}\}$ and then assign each candidate label y_i^j $(j = 1, 2, \ldots, |\mathcal{S}_i|)$ to a replica $\mathbf{x}_i^j$. Eventually, from an original training instance $\mathbf{x}_i$ and its corresponding candidate label set $\mathcal{S}_i$, we can obtain $|\mathcal{S}_i|$ newly generated instance-label pairs, *i.e.*, $(\mathbf{x}_i^1, \mathrm{y}_i^1)$, $(\mathbf{x}_i^2, \mathrm{y}_i^2)$, ..., $(\mathbf{x}_i^{|\mathcal{S}_i|}, \mathrm{y}_i^{|\mathcal{S}_i|})$, and we name these pairs which are generated from the original one instance as a "*multi-birth group*".

After performing the above-mentioned data duplication operation on all training instances, we have transformed the original partially labeled training dataset into a new training dataset which contains $n = \sum_i |\mathcal{S}_i|$ $(i = 1, 2, \ldots, N)$ instances from N multi-birth groups, and meanwhile each instance contains only one label (can be correct or incorrect). It is worth noting that although learning from such transformed dataset is similar to corrupted labels learning [11, 20, 30] at the first glance, it differs from corrupted label learning in that we can definitely know that only one instance is labeled correctly while the labels of other instances are all wrong among each multi-birth group.

As we have obtained the new training dataset, disambiguating the original partially labeled instances is transformed to disambiguating the multi-birth groups, *i.e.*, detecting the unique correctly labeled instance in each multi-birth group. To achieve this target, we take the confidence level of each training instance into consideration. Specifically, we denote $\mathbf{w} = [\mathbf{w}_1^\top, \mathbf{w}_2^\top, \ldots, \mathbf{w}_N^\top]^\top \in \mathbb{R}^{n\times 1}$ as the confidence vector of n training instances from N multi-birth groups, where $\mathbf{w}_i = [w_i^1, w_i^2, \ldots, w_i^{|\mathcal{S}_i|}]^\top$ indicates the group confidence vector of the i-th multi-birth group with the j-th element $w_i^j \in [0, 1]$ in $\mathbf{w}_i$ representing the learning confidence score of the instance $\mathbf{x}_i^j$. As there is only one instance labeled correctly in each multi-birth group, the instances in the same multi-birth group are naturally in a competitive relationship. Therefore, we assume that each group confidence vector should be normalized, *i.e.*, $\sum_{j=1}^{|\mathcal{S}_i|} w_i^j = 1, \forall i = 1, 2, \ldots, N$. Distinctly, disambiguating the multi-birth groups is equivalent to refining their corresponding group confidence vectors.

3.2 Progressive Disambiguation

As stated before, we attempt to disambiguate the simple multi-birth groups at the initial training stages and gradually disambiguate more complicated ones as the training process goes on. That is to say, the group confidence vectors of the simple multi-birth groups ought to be acquired firstly so that the trained model is capable of learning from these disambiguated multi-birth groups. With the proceeding of training process, the disambiguation ability of the model will be improved and thus the group confidence vectors of the complicated multi-birth groups can be obtained precisely.

Intuitively, if a multi-birth group contains an instance which is probably labeled correctly, disambiguating this multi-birth group is relatively easy and thus we consider it as a simple multi-birth group. Existing researches [1, 34] have shown that a network will learn clean and easy patterns firstly, which indicates that the instances with small loss values are likely to be correctly labeled. Based on such observation and meanwhile employing ANNs as the backbone, we propose a progressive disambiguation strategy as explained below.

After feeding the mini-batch data $\mathcal{D}^b$ into the network at the t-th epoch, we can obtain the cross-entropy loss values of these instances, namely $\boldsymbol{\ell}(\Theta, \mathcal{D}^b)$, where Θ indicates the network parameters. After that, we pick up the instances which are likely to be correctly labeled according to the following two conditions: 1) Their loss values are the first $T(t)$ percentage minimums out of $\boldsymbol{\ell}(\Theta, \mathcal{D}^b)$, where $T(t)$ is a time-dependent parameter determining the maximum amount of the simple multi-birth groups at the t-th epoch, and we will introduce it later; and 2) They must be predicted correctly, *i.e.*, the network predictions on them are identical to their labels. After the above screening operation, we can fetch several small-loss instances from the mini-batch $\mathcal{D}^b$ and we regard them as *reliable instances*. It is worth noting that each multi-birth group contains at most one reliable instance because of the constraint from the second condition. Next, we can divide multi-birth groups into two levels of difficulty according to whether they contain a reliable instance, namely simple multi-birth groups and complicated multi-birth groups. Each simple multi-birth group contains one reliable instance which is likely to be correctly labeled, and thus we consider this multi-birth group is relatively easy to disambiguate at the current epoch. Therefore, we disambiguate it by assigning distinguishing confidence scores to the instances among it according to their loss values. If the i-th multi-birth group is a simple multi-birth group, its corresponding group confidence vector $\mathbf{w}_i$ can be updated as:

$$w_i^j = \frac{\exp(-\ell_i^j)}{\sum_{k=1}^{|\mathcal{S}_i|} \exp(-\ell_i^k)}, j = 1, 2, \ldots, |\mathcal{S}_i|, \tag{1}$$

where ℓ_i^j (ℓ_i^k) indicates the loss value of the j-th (k-th) instance in the i-th multi-birth group. Eq. (1) indicates that the instances with small loss values can acquire relatively large confidence scores and meanwhile the normalization constraints of group confidence vectors can be satisfied. As to the complicated multi-birth groups which do not contain any reliable instance, we assign an average confidence vector to them as we cannot figure out the correctly labeled instances among them, namely:

$$w_i^j = \frac{1}{|\mathcal{S}_i|}, j = 1, 2, \ldots, |\mathcal{S}_i|. \tag{2}$$

As no loss value will be generated before the first epoch, all group confidence vectors are initialized in an average manner according to Eq. (2).

After we have obtained the group confidence vector of each multi-birth group, we can clearly know that the instances with large confidence scores are likely to be correctly labeled, and thereby the trained network should pay more attention to them. Otherwise, the network ought to avoid learning from these instances. Taking this into account, we assign weights to the loss values of the instances (*i.e.*, $\boldsymbol{\ell}(\Theta, \mathcal{D}^b)$) with their respective confidence scores, and the propagated back loss of $\mathcal{D}^b$, *i.e.*, $\mathcal{L}(\Theta, \mathcal{D}^b)$, can be calculated as follows:

$$\mathcal{L}(\Theta, \mathcal{D}^b) = {\mathbf{w}^b}^\top \boldsymbol{\ell}(\Theta, \mathcal{D}^b), \tag{3}$$

where $\mathbf{w}^b$ is the confidence vector concatenated by the confidence scores of instances in $\mathcal{D}^b$. Finally, by denoting η as the learning rate, the network parameters Θ can be updated as:

$$\Theta := \Theta - \eta \nabla \mathcal{L}(\Theta, \mathcal{D}^b). \tag{4}$$

As mentioned previously, $T(t)$ is a time-dependent parameter which implies that at most $T(t)$ percentage of multi-birth groups will be regarded as simple multi-birth groups and disambiguated at the t-th epoch, and it will increase from zero to one as the training process proceeds. The concrete formulation of $T(t)$ is as follows:

$$T(t) = \begin{cases} \exp(-5(t/t_r - 1)^2) & t \leq t_r \\ 1 & t > t_r \end{cases}, \tag{5}$$

where t_r is a coefficient determining at which epoch $T(t)$ reaches to one, meaning that almost all the multi-birth groups will be disambiguated after that epoch. Eq. (5) reveals that at the initial training phase, only very few yet simple multi-birth groups will be disambiguated as $T(t)$ is relatively small. With the advance of training steps, the network disambiguation ability will be strengthened and it is capable of disambiguating the complicated multi-birth groups, and thereby $T(t)$ ought to increase accordingly.

The pseudo code of the progressive disambiguation strategy is summarized in Algorithm 1. After initializing the confidence vector $\mathbf{w}$ and feeding the data into the classifier (Steps 1-4), we firstly calculate the widely-used cross-entropy loss of each instance (Step 5). Then, we are able to obtain the reliable instances according to the abovementioned two conditions (Steps 6-8). After that, the confidence vector $\mathbf{w}^b$ will be updated and the corresponding back propagated loss can be calculated (Steps 9-10). Finally, we update $T(t)$ for the next epoch (Step 12).

3.3 Network Cooperation

Although the aforementioned progressive disambiguation strategy has taken the disambiguation difficulty of multi-birth groups into consideration, the corresponding training process is still single-trend of which the disambiguated data will be directly transfered back to the model itself, and the accompanied shortcomings have been analyzed before. Inspired by the work [13, 32] dealing with corrupted label learning problem, we devise a network cooperation mechanism,

Algorithm 1 The Progressive Disambiguation Algorithm

```
Input: $\Theta$, learning rate $\eta$, epoch $t_{max}$, iteration $b_{max}$, training set $\mathcal{D}$
Output: $\Theta$
Initialize $\mathbf{w} = [\mathbf{w}_1; \mathbf{w}_2; \ldots; \mathbf{w}_N]^\top$ according to Eq. (2);
for $t = 1, 2, \ldots, t_{max}$ do
  for $b = 1, 2, \ldots, b_{max}$ do
    Fetch mini-batch $\mathcal{D}^b$ from $\mathcal{D}$;
    Obtain cross-entropy loss values $\boldsymbol{\ell}(\Theta, \mathcal{D}^b)$;
    Obtain first $T(t)$ percentage minimums small-loss instances $\mathcal{D}^s$ from $\mathcal{D}^b$;
    Obtain correctly predicted instances $\mathcal{D}^c$ from $\mathcal{D}^b$;
    Obtain reliable instances $\mathcal{D}^r = \mathcal{D}^s \cap \mathcal{D}^c$;
    Update $\mathbf{w}^b$ according to Eq. (1) and Eq. (2);
    Obtain $\mathcal{L}(\Theta, \mathcal{D}^b)$ according to Eq. (3);
    Update $\Theta$ according to Eq. (4);
    Update $T(t)$ according to Eq. (5);
  end for
end for
return $\Theta$.
```

which trains two networks collaboratively and lets them interact with each other regarding the confidence levels of the instances.

By denoting the two networks as α (with parameter Θ_α) and β (with parameter Θ_β) respectively, we can obtain two confidence vectors of $\mathcal{D}^b$ generated by them independently (according to Section 3.2), *i.e.*, $\mathbf{w}^b_\alpha$ and $\mathbf{w}^b_\beta$. After that, we exchange the confidence vectors among two networks to calculate their respective back propagated loss, *i.e.*, $\mathcal{L}_\alpha(\Theta_\alpha, \mathcal{D}^b)$ and $\mathcal{L}_\beta(\Theta_\beta, \mathcal{D}^b)$:

$$\mathcal{L}_\alpha(\Theta_\alpha, \mathcal{D}^b) = {\mathbf{w}^b_\beta}^\top \boldsymbol{\ell}(\Theta_\alpha, \mathcal{D}^b), \tag{6}$$

$$\mathcal{L}_\beta(\Theta_\beta, \mathcal{D}^b) = {\mathbf{w}^b_\alpha}^\top \boldsymbol{\ell}(\Theta_\beta, \mathcal{D}^b), \tag{7}$$

where $\boldsymbol{\ell}(\Theta_\alpha, \mathcal{D}^b)$ and $\boldsymbol{\ell}(\Theta_\beta, \mathcal{D}^b)$ denote the loss values of the mini-batch $\mathcal{D}^b$ calculated by the network α and network β respectively in the forward propagation phase.

Eq. (6) and Eq. (7) indicate that each network exploits the data disambiguated by its peer network to train itself. As two networks have different ability and can disambiguate multi-birth groups at different levels, exchanging the confidence scores of instances is beneficial for both networks to reduce their respective disambiguation errors, and therefore the error accumulation problem inherited by the conventional single-trend training scheme can be effectively alleviated. Finally, we update the network parameters Θ_α and Θ_β as follows:

$$\Theta_\alpha := \Theta_\alpha - \eta \nabla \mathcal{L}_\alpha(\Theta_\alpha, \mathcal{D}^b), \tag{8}$$

$$\Theta_\beta := \Theta_\beta - \eta \nabla \mathcal{L}_\beta(\Theta_\beta, \mathcal{D}^b). \tag{9}$$

Algorithm 2 The NCPD Algorithm

```
Input: $\Theta_\alpha$, $\Theta_\beta$, learning rate $\eta$, epoch $t_{max}$, iteration $b_{max}$, training set $\mathcal{D}$
Output: $\Theta_\alpha$, $\Theta_\beta$
1: Initialize $\mathbf{w} = [\mathbf{w}_1; \mathbf{w}_2; \ldots; \mathbf{w}_N]^\top$ according to Eq. (2);
2: for $t = 1, 2, \ldots, t_{max}$ do
3:   for $b = 1, 2, \ldots, b_{max}$ do
4:     Update $\mathbf{w}_\alpha^b$ and $\mathbf{w}_\beta^b$ according to Steps 4-9 in Algorithm 1;
5:     Obtain $\mathcal{L}_\alpha(\Theta_\alpha, \mathcal{D}^b)$ and $\mathcal{L}_\beta(\Theta_\alpha, \mathcal{D}^b)$ according to Eq. (6) and Eq. (7) respectively;
6:     Update $\Theta_\alpha$ and $\Theta_\beta$ according to Eq. (8) and Eq. (9) respectively;
7:     Update $T(t)$ according to Eq. (5);
8:   end for
9: end for
10: return $\Theta_\alpha$, $\Theta_\beta$.
```

The pseudo code of the proposed NCPD approach is summarized in Algorithm 2. By employing the progressive disambiguation strategy, we can obtain two different confidence vectors from the two networks, respectively (Step 4). After that, two networks exchange their confidence vectors to calculate the back propagated loss and then update their corresponding parameters (Steps 5-6). Similar to Algorithm 1, the time-dependent parameter $T(t)$ will be updated at each epoch (Step 7).

4 Experiments

4.1 Experimental Setup

In this paper, we conduct comparative experiments to demonstrate the effectiveness of NCPD on two kinds of datasets, *i.e.*, controlled UCI datasets and real-world partial label datasets. The compared state-of-the-art PLL algorithms includes:

- PLKNN [15]: an averaging-based disambiguation approach which generalizes k-nearest neighbor classification for partial label learning;
- M3PL [31]: an identification-based approach that utilizes the maximum margin criterion;
- IPAL [36]: an instance-based approach that employs label propagation procedure to leverage the structural information in feature space;
- SURE [8]: an approach that employs the idea of self-training to exaggerate the mutually exclusive relationships among candidate labels;
- AGGD [27]: an approach that discoveries the manifold structure on original feature space.

For our NCPD approach, we employ the 4-layer perceptron as the backbone and meanwhile utilize Adam [17] to optimize the networks for all experiments.

Table 1: Characteristics and the parameter configurations of the controlled UCI datasets.

Datasets	glass	ecoil	vehicle	abalone
# Instances	214	336	846	4,177
# Features	10	7	18	7
# Labels	5	8	4	29

Configurations:
(I) $r = 1, p \in \{0.1, 0.2, \cdots, 0.7\}$
(II) $r = 2, p \in \{0.1, 0.2, \cdots, 0.7\}$
(III) $r = 3, p \in \{0.1, 0.2, \cdots, 0.7\}$

Besides, we employ the minibatch size of 128 for all runnings and choose the parameter t_r via cross-validation. For baseline methods, they are implemented with parameters setup suggested in respective literatures. Specifically, the regularization parameter C_{max} in M3PL is chosen from the set $\{0.01, 0.1, 1, 10, 100\}$ via cross-validation. In PLKNN, IPAL, and AGGD, the number of nearest numbers k is chosen from set $\{5, 10, 15, 20\}$. Furthermore, we perform ten-fold cross-validation to record the mean prediction accuracies and standard deviations for all comparing algorithms on all the datasets adopted below.

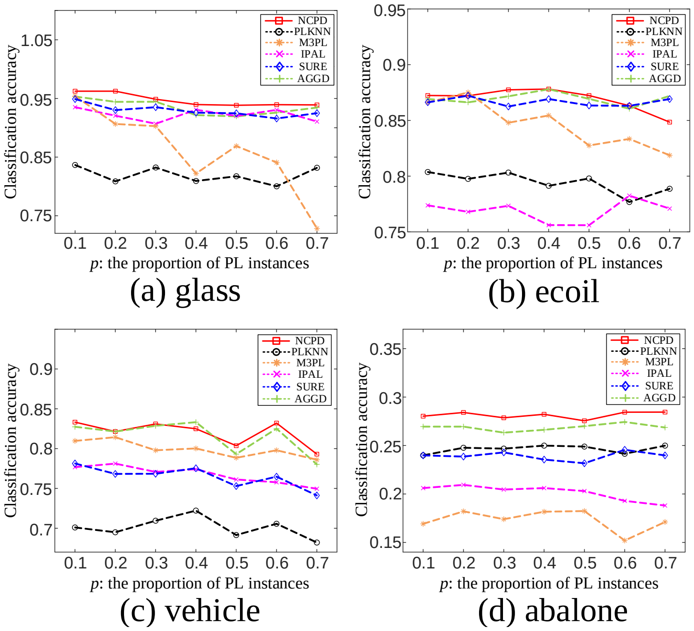
4.2 Experiments on Controlled UCI Datasets

Following the widely-used controlling protocol in previous PLL works [7, 25, 28, 36–38], an artificial partial label dataset can be generated from an original UCI dataset with two controlling parameters p and r. To be specific, p controls the proportion of instances which are partially labeled (*i.e.*, $|S_i| > 1$), and r controls the number of false positive labels in each candidate label set (*i.e.*, $|S_i| = r + 1$). The characteristics of these controlled UCI datasets as well as the parameter configurations are listed in Table 1.

Fig. 3, Fig. 4, and Fig. 5 show the classification accuracy of each algorithm as p ranges from 0.1 to 0.7 with the step size 0.1, when $r = 1$, $r = 2$, and $r = 3$ (Configuration (I), (II), and (III)), respectively. As illustrated in these figures, NCPD achieves superior performance against other comparing algorithms on these controlled UCI datasets. Specifically, NCPD achieves superior or at least comparable performance against PLKNN, M3PL, and IPAL in all experiments. As to SURE and AGGD, although their classification accuracies are slightly higher than NCPD in a few parameter configurations, they are inferior to NCPD in most cases.

4.3 Experiments on Real-world Datasets

Apart from the controlled UCI datasets, we also conduct experiments on five real-world partial label datasets which are collected from several application domains



(a) glass (b) ecoil

(c) vehicle (d) abalone

Fig. 3: Classification accuracy of each algorithm on controlled UCI datasets with p ranging from 0.1 to 0.7 ($r = 1$).

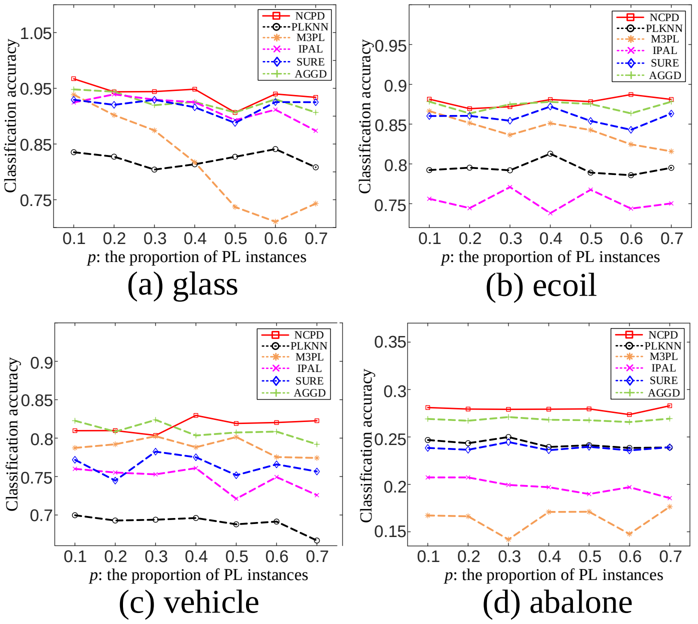
Table 2: Characteristics of adopted real-world partial label datasets.

Datasets	Lost	BirdSong	MSRCv2	Soccer Player	Yahoo!News
# Instances	1,122	4,998	1,758	17,472	22,991
# Features	108	38	48	279	163
# Labels	16	13	23	171	219
# Avg. CLs	2.23	2.18	3.16	2.09	1.91

including *Lost* [6], *Soccer Player* [33], and *Yahoo!News* [12] for automatic face naming, *MSRCv2* [19] for object classification, and *BirdSong* [3] for bird song classification. The characteristics of these real-world datasets are summarized in Table 2 where the average number of candidate labels of each dataset (*i.e.*, # Avg. CLs) is also reported[4].

The average classification accuracies as well as the standard deviations of different approaches on these real-world datasets are shown in Table 3. Pairwise t-test at 0.05 significance level is also conducted based on the results of ten-fold cross-validation. From Table 3, we have three findings: 1) NCPD achieves the highest classification accuracies among all baselines on all adopted real-world

[4] These datasets are available at http://palm.seu.edu.cn/zhangml.



(a) glass

(b) ecoil

(c) vehicle

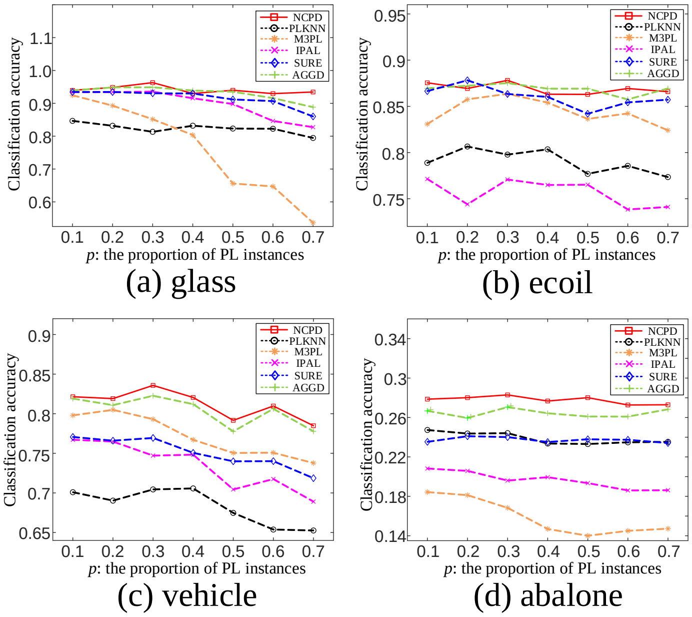
(d) abalone

Fig. 4: Classification accuracy of each algorithm on controlled UCI datasets with p ranging from 0.1 to 0.7 ($r = 2$).

Table 3: Classification accuracy (mean ± std) of each algorithm on five real-world datasets. •/◦ indicates that NCPD is significantly superior / inferior to the comparing algorithm on the corresponding dataset (pairwise t-test with 0.05 significance level).

	Lost	BirdSong	MSRCv2	Soccer Player	Yahoo!News
PLKNN	0.471 ± 0.032 •	0.686 ± 0.015 •	0.457 ± 0.049 •	0.530 ± 0.016 •	0.482 ± 0.011 •
M3PL	0.721 ± 0.037 •	0.667 ± 0.042 •	0.474 ± 0.038 •	0.500 ± 0.007 •	0.628 ± 0.013 •
IPAL	0.653 ± 0.022 •	0.734 ± 0.013 •	0.537 ± 0.045 •	0.547 ± 0.016 •	0.577 ± 0.010 •
SURE	0.739 ± 0.036 •	0.730 ± 0.015 •	0.508 ± 0.043 •	0.522 ± 0.013 •	0.562 ± 0.011 •
AGGD	0.778 ± 0.040	0.737 ± 0.018	0.506 ± 0.041 •	0.543 ± 0.016 •	0.637 ± 0.008 •
NCPD	**0.790 ± 0.055**	**0.751 ± 0.018**	**0.589 ± 0.046**	**0.573 ± 0.013**	**0.657 ± 0.013**

datasets; 2) NCPD significantly outperforms PLKNN, M3PL, IPAL, and SURE on all these datasets; 3) NCPD is never statistically inferior to any comparing algorithms in all cases. These findings convincingly substantiate the superiority of our NCPD approach to other comparators.


(a) glass

(b) ecoil

(c) vehicle

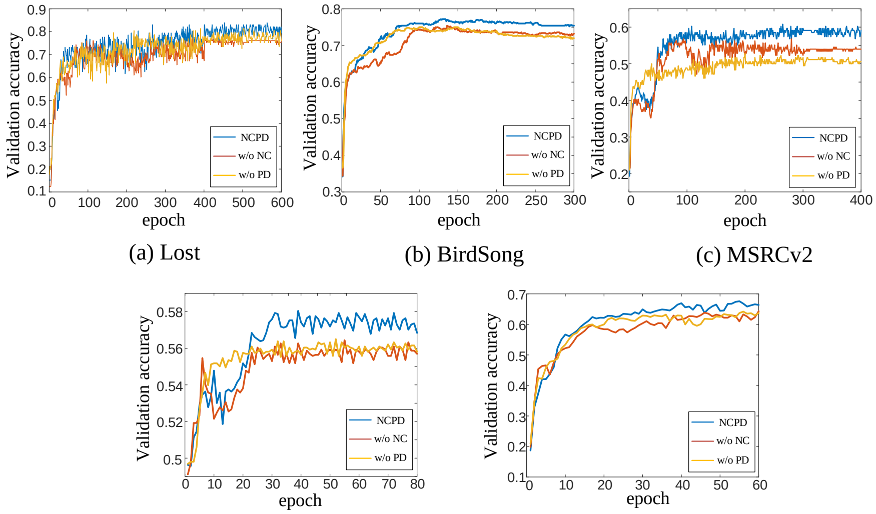
(d) abalone

Fig. 5: Classification accuracy of each algorithm on controlled UCI datasets with p ranging from 0.1 to 0.7 ($r = 3$).

4.4 Ablation Study

The superiority of the proposed NCPD approach has been verified by thorough experimental results presented above. In this section, we conduct ablation study on adopted real-world datasets to further demonstrate the effectiveness of the two crucial techniques employed by NPCD, *i.e.*, the progressive disambiguation strategy and the network cooperation mechanism.

Specifically, to demonstrate the effectiveness of the progressive disambiguation strategy, we discard this strategy and merely train two networks with network cooperation mechanism, *i.e.*, all multi-birth groups are disambiguated according to Eq. (1) in every epoch regardless their disambiguation difficulty. To confirm the effectiveness of the network cooperation mechanism, we barely train one network equipped with the progressive disambiguation strategy (see Section 3.2). Fig. 6 shows the results, from which we can observe that the integrated NCPD approach generates the highest accuracies than other two settings (*i.e.*, "w/o NC" and "w/o PD"). In contrast, the accuracies will decrease when either the progressive disambiguation strategy or the network cooperation mechanism is removed, therefore the effectiveness and indispensability of these two crucial techniques are validated.



(a) Lost

(b) BirdSong

(c) MSRCv2

(d) Soccer Player

(e) Yahoo!News

Fig. 6: Validation accuracy with different settings on adopted real-world datasets. The blue curve denotes the accuracy of the integrated NCPD approach (legend by "NCPD"). The red curve and the yellow curve indicate the accuracy of NCPD that removes the network cooperation mechanism (denoted by "w/o NC") and the progressive disambiguation strategy (denoted by "w/o PD"), respectively.

5 Conclusion

In this paper, we propose a novel approach for PLL which is dubbed as "NCPD". By employing the progressive disambiguation strategy, our approach is able to exploit the disambiguation difficulty of the instances and then disambiguate them in a progressive manner, which is beneficial for the steady improvement of model capability and thereby the adverse impacts brought by false positive labels can be effectively reduced. Furthermore, the network cooperation mechanism greatly facilitates the salutary mutual learning process between two networks, and therefore can effectively alleviate the error accumulation problem inherited by the existing single-trend training framework. Thorough experimental results on various datasets demonstrate the effectiveness of the proposed NCPD approach. Considering that how to determine the disambiguation difficulty of the instances plays a vital role in our algorithm, we will devise a more advanced methodology to judge the disambiguation difficulty of these partially labeled instances in the future.

Acknowledgments

This research is supported by NSF of China (Nos: 61973162, U1713208), NSF of Jiangsu Province (No: BK20171430), the Fundamental Research Funds for the Central Universities (No: 30920032202), the "Young Elite Scientists Sponsorship Program" by Jiangsu Province and CAST (No: 2018QNRC001), and the Program for Changjiang Scholars.

References

1. Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M.S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al.: A closer look at memorization in deep networks. In: Proc. International Conference on Machine Learning. pp. 233–242 (2017)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Eleventh Annual Conference on Computational Learning Theory. pp. 92–100 (1998)
3. Briggs, F., Fern, X.Z., Raich, R.: Rank-loss support instance machines for miml instance annotation. In: Proc. International Conference on Knowledge Discovery and Data Mining. pp. 534–542 (2012)
4. Chen, C.H., Patel, V.M., Chellappa, R.: Learning from ambiguously labeled face images. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(7), 1653–1667 (2018)
5. Chen, Z.S., Wu, X., Chen, Q.G., Hu, Y., Zhang, M.L.: Multi-view partial multi-label learning with graph-based disambiguation. In: Proc. AAAI Conference on Artificial Intelligence. pp. 3553–3560 (2020)
6. Cour, T., Sapp, B., Jordan, C., Taskar, B.: Learning from ambiguously labeled images. In: Proc. Computer Vision and Pattern Recognition. pp. 919–926 (2009)
7. Feng, L., An, B.: Leveraging latent label distributions for partial label learning. In: Proc. International Joint Conference on Artificial Intelligence. pp. 2107–2113 (2018)
8. Feng, L., An, B.: Partial label learning with self-guided retraining. In: Proc. AAAI Conference on Artificial Intelligence. pp. 3542–3549 (2019)
9. Gong, C., Liu, T., Tang, Y., Yang, J., Yang, J., Tao, D.: A regularization approach for instancebased superset label learning. IEEE Transactions on Cybernetics **48**(3), 967–978 (2018)
10. Gong, C., Shi, H., Liu, T., Zhang, C., Yang, J., Tao, D.: Loss decomposition and centroid estimation for positive and unlabeled learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
11. Gong, C., Zhang, H., Yang, J., Tao, D.: Learning with inadequate and incorrect supervision. In: Proc. International Conference on Data Mining. pp. 889–894 (2017)
12. Guillaumin, M., Verbeek, J., Schmid, C.: Multiple instance metric learning from automatically labeled bags of faces. In: Proc. European Conference on Computer Vision. pp. 634–647 (2010)
13. Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., Sugiyama, M.: Co-teaching: Robust training of deep neural networks with extremely noisy labels. In: Proc. Advances in Neural Information Processing Systems. pp. 8527–8537 (2018)
14. Hassoun, M.H., et al.: Fundamentals of artificial neural networks. MIT press (1995)

15. Hüllermeier, E., Beringer, J.: Learning from ambiguously labeled examples. Intelligent Data Analysis **10**(5), 419–439 (2006)
16. Jin, R., Ghahramani, Z.: Learning with multiple labels. In: Proc. Advances in Nural Information Processing Systems. pp. 921–928 (2003)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Liu, L., Dietterich, T.: Learnability of the superset label learning problem. In: Proc. International Conference on Machine Learning. pp. 1629–1637 (2014)
19. Liu, L., Dietterich, T.G.: A conditional multinomial mixture model for superset label learning. In: Proc. Advances in Neural Information Processing Systems. pp. 548–556 (2012)
20. Liu, T., Tao, D.: Classification with noisy labels by importance reweighting. IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(3), 447–461 (2015)
21. Liu, W., Xu, D., Tsang, I.W., Zhang, W.: Metric learning for multi-output tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(2), 408–422 (2018)
22. Luo, J., Orabona, F.: Learning from candidate labeling sets. In: Proc. Advances in Neural Information Processing Systems. pp. 1504–1512 (2010)
23. Lyu, G., Feng, S., Lang, C., Wang, T.: A self-paced regularization framework for partial-label learning. arXiv preprint arXiv:1804.07759 (2018)
24. Nguyen, N., Caruana, R.: Classification with partial labels. In: Proc. International Conference on Knowledge Discovery and Data Mining. pp. 551–559 (2008)
25. Tang, C.Z., Zhang, M.L.: Confidence-rated discriminative partial label learning. In: Proc. AAAI Conference on Artificial Intelligence (2017)
26. Wan, S., Gong, C., Zhong, P., Du, B., Zhang, L., Yang, J.: Multiscale dynamic graph convolutional network for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing **58**(5), 3162–3177 (2019)
27. Wang, D.B., Li, L., Zhang, M.L.: Adaptive graph guided disambiguation for partial label learning. In: Proc. International Conference on Knowledge Discovery and Data Mining. pp. 83–91 (2019)
28. Wu, X., Zhang, M.L.: Towards enabling binary decomposition for partial label learning. In: Proc. International Joint Conference on Artificial Intelligence. pp. 2868–2874 (2018)
29. Yao, Y., Deng, J., Chen, X., Gong, C., Wu, J., Yang, J.: Deep discriminative cnn with temporal ensembling for ambiguously-labeled image classification. In: Proc. AAAI Conference on Artificial Intelligence. pp. 12669–12676 (2020)
30. Yi, K., Wu, J.: Probabilistic end-to-end noise correction for learning with noisy labels. In: Proc. Computer Vision and Pattern Recognition. pp. 7017–7025 (2019)
31. Yu, F., Zhang, M.L.: Maximum margin partial label learning. Machine Learning **106**(4), 573–593 (2017)
32. Yu, X., Han, B., Yao, J., Niu, G., Tsang, I.W., Sugiyama, M.: How does disagreement help generalization against label corruption? arXiv preprint arXiv:1901.04215 (2019)
33. Zeng, Z., Xiao, S., Jia, K., Chan, T.H., Gao, S., Xu, D., Ma, Y.: Learning by associating ambiguously labeled images. In: Proc. Computer Vision and Pattern Recognition. pp. 708– 715 (2013)
34. Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O.: Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530 (2016)
35. Zhang, C., Ren, D., Liu, T., Yang, J., Gong, C.: Positive and unlabeled learning with label disambiguation. In: Proc. International Joint Conference on Artificial Intelligence. pp. 4250– 4256 (2019)

36. Zhang, M.L., Yu, F.: Solving the partial label learning problem: An instance-based approach. In: Proc. International Joint Conference on Artificial Intelligence. pp. 4048–4054 (2015)
37. Zhang, M.L., Yu, F., Tang, C.Z.: Disambiguation-free partial label learning. IEEE Transactions on Knowledge and Data Engineering **29**(10), 2155–2167 (2017)
38. Zhang, M.L., Zhou, B.B., Liu, X.Y.: Partial label learning via feature-aware disambiguation. In: Proc. International Conference on Knowledge Discovery and Data Mining. pp. 1335– 1344 (2016)