Revisiting Consistency Regularization for Deep Partial Label Learning

Dong-Dong Wu [1 2 †] **Deng-Bao Wang** [1 2 †] **Min-Ling Zhang** [1 2 *]

Abstract

Partial label learning (PLL), which refers to the classification task where each training instance is ambiguously annotated with a set of candidate labels, has been recently studied in deep learning paradigm. Despite advances in recent deep PLL literature, existing methods (e.g., methods based on self-training or contrastive learning) are confronted with either ineffectiveness or inefficiency. In this paper, we revisit a simple idea namely *consistency regularization*, which has been shown effective in traditional PLL literature, to guide the training of deep models. Towards this goal, a new regularized training framework, which performs supervised learning on non-candidate labels and employs consistency regularization on candidate labels, is proposed for PLL. We instantiate the regularization term by matching the outputs of multiple augmentations of an instance to a conformal label distribution, which can be adaptively inferred by the closed-form solution. Experiments on benchmark datasets demonstrate the superiority of the proposed method compared with other state-of-the-art methods.

1. Introduction

Collecting large scale datasets with high-quality annotations for supervised learning algorithm is generally difficult. To overcome this problem, weakly supervised learning has been widely studied in recent years, which includes, but is not limited to, multi-label learning (Zhang & Zhou, 2013; Liu et al., 2020; Xu & Guo, 2021), noisy-label learning (Natarajan et al., 2013; Wang et al., 2021), positive-unlabeled learning (Kiryo et al., 2017; Su et al., 2021), and semi-supervised learning (Zhu & Goldberg, 2009; Xu et al., 2021b). This paper focuses on a typical weakly supervised learning problem known as partial label learning (PLL), which refers to the task where each training instance is annotated with a set of candidate labels, among which only one is the ground-truth. This problem naturally arises in various real-world applications, such as web mining (Luo & Orabona, 2010), multimedia content analysis (Zeng et al., 2013), and automatic image annotations (Chen et al., 2017).

The major difficulty of PLL lies in the ambiguity of label space, since the ground-truth label is concealed in its candidate label set and not directly accessible to the learning algorithm. Traditional PLL methods tackle this problem by conducting label disambiguation before or during training. For example, Zhang & Yu (2015) solve the PLL problem by firstly employing label propagation algorithm to disambiguate the partial labels, and then learning a classification model on the processed dataset. In (Jin & Ghahramani, 2002; Liu & Dietterich, 2012), the ground-truth label of each instance is regarded as a latent variable which is identified via the EM algorithm. Furthermore, *manifold consistency regularization*, which assumes that the manifold structure in the feature space should also be preserved in the label space, has been shown very effective to help disambiguate the candidate labels during training (Zhang et al., 2016; Feng & An, 2018; Wang et al., 2019). However, these methods are usually restricted to linear or kenel-based models and inefficient when dealing with large scale datasets and hign-dimensional natural features.

As deep neural networks become popular, PLL has been explosively studied in deep learning paradigm recently. Based on the uniform partial label generation assumption, Feng et al. (2020) derive two methods which are risk-consistent and classifier-consistent respectively. Based on the instance-dependent partial label generation assumption, Xu et al. (2021a) propose to recover the latent label distribution with the variational inference technique. In (Lv et al., 2020; Wen et al., 2021), self-training technique is adopted in deep learning framework to progressively identify the ground-truth labels during training. These methods can be efficiently trained with stochastic optimization and agnostic in specific network architectures. However, severe error accumulation problem may be caused due to the low quality prediction in initial training stage. A recent research (Wang et al., 2022) adopts the idea of unsupervised contrastive representation

† Equal contribution [1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China [2]Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. * Correspondence to: Min-Ling Zhang <zhangml@seu.edu.cn>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

learning to PLL and achieves very impressive results on several image classification benchmarks. Nevertheless, the contrastive learning procedure is usually time-consuming and resource-demanding. Overall, existing deep PLL methods are confronted with either ineffectiveness or inefficiency (e.g., methods based on self-training or contrastive learning); Moreover, they usually assume that partial labels come from a specific generation process and hence cannot handle all PLL scenarios.

Inspired by the success of manifold consistency regularization in traditional PLL literature, we revisit the utilization of consistency regularization to guide the design of deep learning method in this paper. Towards this goal, a concise regularized training framework, which performs supervised learning on non-candidate labels and employs consistency regularization on candidate labels, is proposed for PLL. We instantiate the regularization term by matching the outputs of multiple augmentations of each instance to a conformal label distribution, which can be adaptively inferred by the proposed closed-form solution in bi-level optimization framework. To guarantee robust network training on high-ambiguity partial label datasets, we propose to gradually induce the regularization during training by equipping a dynamic balancing factor in the unified objective function. Moreover, the proposed method does not make any assumption on the partial label generation process, and hence can deal with both uniform and instance-dependent partial labels. Our contributions can be summarized as follows:

- A new regularized training framework is proposed for deep PLL. To the best of our knowledge, our work explores the consistency regularization in deep PLL literature for the first time.
- We propose a powerful regularization term by involving the conformal label distribution, which could be adaptively inferred during network training in the bi-level optimization framework.
- Experimental results on both uniform and instance-dependent partial label datasets demonstrate the effectiveness of our concise method compared with the current state-of-the-art methods.

The rest of this paper is organized as follows. We review related work in Section 2, and introduce our method in Section 3; Section 4 presents the experimental results, followed by the conclusion in Section 5.

2. Related Work

2.1. Traditional Partial Label Learning

As the principal approach in traditional PLL literature, label disambiguation can be achieved based on *averaging* or *identification* strategy. Averaging-based methods distinguish candidate labels from the non-candidate ones while treats all the candidate labels equally (Hüllermeier & Beringer, 2006; Cour et al., 2011; Zhang & Yu, 2015). Identification-based strategy progressively refines the confidence of each individual candidate label by treating it as a latent variable (Jin & Ghahramani, 2002; Yu & Zhang, 2016). A wealth of popular learning techniques have been adopted to the identification-based label disambiguation framework. For example, Jin & Ghahramani (2002) use maximum likelihood criterion to learn the classier from the unknown target distribution, which is iteratively inferred from the classifier itself. Yu & Zhang (2016) employ the maximum margin constraint in PLL problem, where the margin between the maximum modeling output from candidate labels and that from other labels is optimized. Generally speaking, identification-based methods are more effective than averaging-based methods and have attracted much more research interest in the past decades (Jin & Ghahramani, 2002; Nguyen & Caruana, 2008; Liu & Dietterich, 2012; Feng & An, 2019; Ni et al., 2021).

Manifold consistency regularization, which assumes the manifold structure in feature space should also be preserved in label space, has been shown work well in PLL problem and is usually considered as a specific type of identification-based label disambiguation methods. Zhang et al. (2016) and Wu et al. (2020) solve the PLL problem by firstly using manifold learning techniques to disambiguate partial labels, and then learning a classification model on the processed dataset. In several later studies, the manifold consistency assumption is wrapped as a regularization term in the objective function and hence can be flexibly accomplished during model training (Gong et al., 2017; Feng & An, 2018; Wang et al., 2019).

2.2. Deep Partial Label Learning

Traditional methods are restricted to linear models, which are usually optimized in low-efficiency manners. As deep neural networks become popular, PLL has been recently studied in deep learning paradigm, in which the PLL learner is compatible with highly efficient stochastic optimization. Yao et al. (2020a) conduct a preliminary study on deep PLL, in which a temporal-ensembling technique is adopted in the training of deep neural networks; In (Yao et al., 2020b), they also propose to utilize a network-cooperation mechanism to train two networks collaboratively and let them interact with each others for reducing the disambiguation errors. Lv et al. (2020) use a simple self-training like strategy to progressively identify the ground-truth labels during network training, while impressive results on image classification benchmarks was achieved for the first time in PLL literature. Wen et al. (2021) introduce a family of loss functions named *leveraged weighted* loss, which takes the method in (Lv et al., 2020) as one of its special cases.

Most existing methods assume that partial labels come from the uniform generation model. Feng et al. (2020) for the first time provide an mathematical formulation for this uniform generation process and derive a risk-consistent method and a classifier-consistent, which can be easily used for training deep neural networks. Furthermore, in many real-world scenario, partial labels can also be generated with the instance-dependent process. Based on this assumption, Yan & Guo (2021) propose a GAN-based method which simultaneously performs label disambiguation with a generative network and maps the instances to the disambiguated labels with a classification network; Xu et al. (2021a) employ variational inference in the network training to adaptively recover the latent label distribution.

Recently, Wang et al. (2022) adopt contrastive representation learning to PLL and significantly improve the state-of-the-art results on several image classification benchmarks. However, the contrastive learning procedure is usually time-consuming and resource-demanding. To deployment learning system in real-world setting, it is desire to design simple yet effective method for training deep neural networks from large-scale datasets, and relax the assumption of partial label generation process. Towards this goal, we next investigate the consistency regularization for deep PLL.

3. Methodology

In this section, we firstly introduce the preliminaries in Subsection 3.1. Then we present our regularized training framework in Subsection 3.2 and the consistency regularization term in Subsection 3.3. Finally we describe the detailed implementations in Subsection 3.4.

3.1. Preliminaries

Notations. Let $\mathcal{X} \subset \mathbb{R}^q$ denote the q-dimensional feature space and $\mathcal{Y} = \{1, 2, ..., c\}$ denote the label space with c distinct labels. PLL assumes that the ground-truth label $y \in \mathcal{Y}$ of instance $\boldsymbol{x} \in \mathcal{X}$ is concealed in its candidate label set $\mathcal{S} \subset \mathcal{Y}$, and its goal is to learn a multi-class classifier $g : \mathcal{X} \to \mathcal{Y}$ that minimizes the classification risk on partial label dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \mathcal{S}_i) | 1 \le i \le n\}$. For the classifier g, we use $g_k(\boldsymbol{x})$ to denote the output of classifier g on label k given input $\boldsymbol{x}$.

Consistency Regularization for Traditional PLL. Manifold consistency assumes that the manifold structure in feature space should also be preserved in the label space. This property is very useful for label disambiguation, and hence has been employed in previous traditional PLL methods. Generally speaking, one can formulate a specific manifold consistency regularization term in the objective function. For example, based on the *locally linear embedding* tech-

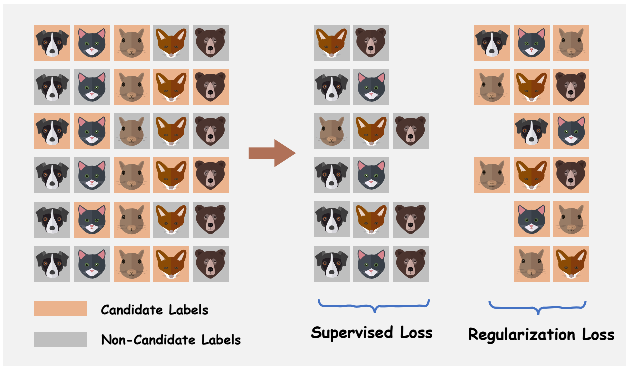


Figure 1. The label matrix is decoupled into two complementary parts including candidate labels and non-candidate labels, on which the supervised loss and regularization loss could be calculated respectively.

nique, the regularization term on instance $\boldsymbol{x}_j$ can be formally written as:

$$\zeta(\boldsymbol{x}_j) = \left\| \boldsymbol{x}_j - \textstyle\sum_i w_{ij} \boldsymbol{x}_i \right\| + \left\| f(\boldsymbol{x}_j) - \textstyle\sum_i w_{ij} f(\boldsymbol{x}_i) \right\|, \quad (1)$$

where w_{ij} is an element of weighting matrix W and represents the construction weight from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$, $f(x_i)$ is a normalized real-valued vector and each $f_j(x_i)$ represents the labeling confidence of the j-th label being the ground-truth label for x_i . The weighting matrix W can be optimized together with model parameters by alternative optimization algorithms (Wang et al., 2019).

Although this consistency regularization term has been empirically shown effective in traditional PLL literature, it is hard to be directly employed in deep learning framework. On the one hand, the manifold structure in raw feature space may be inaccurate to reflect the structure in semantic space when dealing with natural features like RGB values of images. On the other hand, there are additional parameters (e.g., weighting matrix W) to be optimized with some specifically designed algorithms, which may be incompatible with the gradient-based optimization procedure.

3.2. The Overall Framework

The popular regularized training methods learn predictive models from the objective that usually consists of a supervised loss term as well as an additional regularization term. For example, the state-of-the-art semi-supervised learning methods achieve comparable results compared with fully supervised learning by optimizing both the supervised loss on labeled data and the regularization loss on unlabeled data (Sohn et al., 2020; Xie et al., 2020).

Following this framework, we need to firstly obtain the supervised loss, yet directly optimizing the classification loss on these ambiguous labels would lead the bias caused by the false positive labels. Nevertheless, one can extracts perfectly accurate supervision from the non-candidate labels, i.e., the complements of the candidate labels would never be the ground-truth labels. Therefore, we instantiate the supervised loss term with a monotonically increasing function of the outputs on these non-candidate labels. Specifically, given instance $\boldsymbol{x}$ and its candidate label set $\mathcal{S}$, we calculate the following modified negative log likelihood loss:

$$\mathcal{L}_{\texttt{Sup}}(\boldsymbol{x}, \mathcal{S}) = -\sum\nolimits_{k\notin\mathcal{S}} \log(1 - g_k(\boldsymbol{x})). \tag{2}$$

The underlying principle behind this modified loss function is quite simple: *Given that the confidences assigned to non-candidate labels are always zero, the log likelihoods on the outputs of these labels should be as small as possible.* Despite its simple form, it has been empirically shown that optimizing this loss on incorrect labels gives reasonable results (Chou et al., 2020; Gao & Zhang, 2021). After obtaining the supervised loss on the non-candidate labels, we further calculate the regularization loss on the candidate labels. The overall objective function on each instance can be presented as:

$$\mathcal{L}(\boldsymbol{x}, \mathcal{S}) = \mathcal{L}_{\texttt{Sup}}(\boldsymbol{x}, \mathcal{S}) + \lambda\boldsymbol{\Psi}(\boldsymbol{x}, \mathcal{S}), \tag{3}$$

where the multiplicative factor λ is used to balance the contributions of these two loss terms. We instantiate the regularization term $\boldsymbol{\Psi}(\boldsymbol{x}, \mathcal{S})$ by enabling the consistency of model outputs for multiple augmentations of an instance on its candidate labels.

The overall objective can be efficiently optimized in batch-wise training procedure. As is shown in Figure 1, in each batch training, the label matrix would be decoupled into two complementary parts, which could be easily implemented by the masking process, and the supervised loss and regularization loss would be calculated on them respectively. In the next subsection, we will present the details of this regularization term.

3.3. Consistency Regularization on Candidate Labels

With the help of consistency regularized training, we expect to encourage the network's output to be invariant to small changes applied to the feature space. A simple strategy towards this goal is aligning the output distribution of different augmentations of each instance. This can be implemented by minimizing the divergence of each output pair given a set of random augmented instances in $\mathcal{A}(\boldsymbol{x}) = \{\texttt{Aug}_i(\boldsymbol{x}) | 1 \leq i \leq K\}$, where $\texttt{Aug}(\boldsymbol{x})$ denotes a random augmentation of the original instance. However, there may be some random augmentations which could cause significant semantic shift, and simply aligning the outputs of these augmentations would degrade the model performance.

To avoid these above issues, we hypothesis that there exists the conformal label distribution for each instance $\boldsymbol{x}$ that can

Algorithm 1 Our Regularized Training Method

```
Input: Training dataset $\mathcal{D} = \{x_i, S_i\}_{i=1}^n$;
       The classifier $\boldsymbol{g}$ and its initial parameters $\boldsymbol{\theta}$;
       Epochs $T$ and iterations $I$;
       The number of augmentations $K$;
       Maximum balancing factor $\lambda$;
Procedure:
1: Initialize $\boldsymbol{p}$ for each instance by Eq.(6).
2: for $t = 1$ to $T$ do
3:   for $i = 1$ to $I$ do
4:     Fetch a random batch $\mathcal{B}$ from $\mathcal{D}$;
5:     Obtain the conformal label distributions by Eq.(7);
6:     Calculate the loss on the current batch by Eq. (8);
7:     Update network parameter $\boldsymbol{\theta}$ via gradient descent;
8:   end for
9: end for
Output: Learned multi-class classifier $\boldsymbol{g}$.
```

be used to appropriately guide the consistency training of all the augmentations in $\mathcal{A}(\boldsymbol{x})$. Denote the conformal label distribution of a given instance $\boldsymbol{x}$ as $\boldsymbol{p}$, which satisfies that $\sum_{k\in\mathcal{S}} p_k = 1$ and $p_k = 0, \forall k \notin \mathcal{S}$. Now we implement the regularization term by minimizing the Kullback-Leibler (KL) divergence of $\boldsymbol{p}$ and the outputs of all the augmentations in $\mathcal{A}(\boldsymbol{x})$:

$$\boldsymbol{\Psi}(\boldsymbol{x}, \mathcal{S}) = \sum\nolimits_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \texttt{KL}(\boldsymbol{p} || g(\boldsymbol{z})). \tag{4}$$

Bi-level optimization. Given this regularization term, we need to obtain the conformal label distribution of each instance before optimizing the network's parameters. For a specific example $(\boldsymbol{x}, \mathcal{S})$, we suggest that its conformal label distribution $\boldsymbol{p}$ could be treated as latent variable and optimized simultaneously with the parameters $\boldsymbol{\theta}$ from the objective function. By taking the overall objective $\mathcal{L}$ as the function of $\boldsymbol{\theta}$ and $\boldsymbol{p}$, we set up the following bi-level optimization problem:

$$\begin{aligned} &\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}^*) \\ &\text{subject to } \boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}); \\ &\qquad\qquad \sum\nolimits_{k\in\mathcal{S}} p_k = 1;\ p_k = 0, \forall k \notin \mathcal{S}. \end{aligned} \tag{5}$$

Bi-level optimization problem has been proved strongly NP-hard (Jeroslow, 1985), so getting such an exact solution for problem (5) is impossible in polynomial time. Fortunately, the inner optimization problem in (5) has the closed-form solution, which will be presented in Subsection 3.4. Therefore, we use the alternative optimization strategy to approximately solve this problem, in which each conformal label distribution $\boldsymbol{p}$ could be updated in every optimization iteration in an on-the-fly manner. Furthermore, since $\boldsymbol{p}$ reflects the label distribution for all augmentations in $\mathcal{A}(\boldsymbol{x})$, the optimization of this inner problem could be considered as the explicit label disambiguation process for $\boldsymbol{x}$ and work in the transductive PLL setting.

The overall procedure of our method is presented in Algorithm 1. In the beginning, we need to initialize the conformal label distribution $\boldsymbol{p}$ for each example $(\boldsymbol{x}, \mathcal{S})$ by normalizing the candidate labels:

$$p_k = \begin{cases} \frac{1}{|\mathcal{S}|} & \text{if } k \in \mathcal{S}, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Then, in each training iteration, our method firstly obtains the conformal label distributions for all instance in current batch, then calculates the overall loss and lastly updates the network parameters via gradient descent.

3.4. Practical Implementations

Closed-form solution of $\boldsymbol{p}^*$. We firstly come up with the following proposition on the convexity of the loss function with respect to $\boldsymbol{p}$.

Proposition 1. *The loss function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p})$ is convex in the distribution $\boldsymbol{p}$, i.e.,*

$$\mathcal{L}(\boldsymbol{\theta}, \alpha\boldsymbol{p}_1 + (1-\alpha)\boldsymbol{p}_2) \leq \alpha\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_1) + (1-\alpha)\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_2),$$

where $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are two distributions and $0 \leq \alpha \leq 1$.

By taking into account the constrains of problem (5), we can obtain the optimal distribution by using the Lagrangian Multiplier method. As the result, each element of $\boldsymbol{p}^*$ on the candidate labels can be easily calculated as:

$$p_k^* = \frac{\left(\prod_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} g_k(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}}}{\sum_{j\in\mathcal{S}} \left(\prod_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} g_j(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}}}, \tag{7}$$

where $|\mathcal{A}(\boldsymbol{x})|$ means the cardinality of $\mathcal{A}(\boldsymbol{x})$. The detailed derivation of this equation could be found in the Appendix A.2.

Note that with fixed $\boldsymbol{\theta}$, the conformal label distribution $\boldsymbol{p}$ for a specific instance $\boldsymbol{x}$ is independent from each other. Thus, we focus on the optimization of a specific conformal label distribution $\boldsymbol{p}$. In practice, deep neural networks are usually learned with batch-wise training procedure, thus the optimization of all conformal label distributions in each batch could be efficiently conducted with matrix operations.

Gradually Induced Regularization. In objective function (3), we use a balancing factor λ to control the strength of regularization. We suggest that using a fixed balancing factor during the whole training procedure is not a good choice. In the beginning stage, strong regularization may degrade the performance due to the low quality of inferred conformal label distribution. Moreover, as the conformal label distribution becomes more accurate after some training epochs, stronger regularization would always benefit the training. Therefore, we apply a dynamic balancing function in the objective:

$$\mathcal{L}(\boldsymbol{x}, \mathcal{S}) = \mathcal{L}_{\texttt{Sup}}(\boldsymbol{x}, \mathcal{S}) + \gamma(t) \cdot \boldsymbol{\Psi}(\boldsymbol{x}, \mathcal{S}), \tag{8}$$

where λ is replaced by a non-decreasing balancing function with respect to the epoch number t, i.e.,

$$\gamma(t) = \min\{\frac{t}{T'}\lambda, \lambda\}. \tag{9}$$

This dynamic balancing strategy employs small weighting factor in the initial training stage and gradually increasing it during training epochs. Specifically, the factor would be increased to λ at epoch T', then it keeps the constant λ until the end of training.

Data Augmentation. Data augmentation plays an important role in the proposed regularization method. Intuitively speaking, different augmentations should be distinct from the original instance while retaining the semantic information. Among various effectual augmentation techniques, we adopt two commonly used ones: Autoaugment (Cubuk et al., 2019) and Cutout(DeVries & Taylor, 2017). For Autoaugment, we simply use the augmentation policies released by Cubuk et al. (2019). These policies are automatically searched from Python Image Library (PIL) and have been empirically shown promising on multiple downstream image recognition tasks. In our implementation, we firstly concatenate these policies into a pool, then randomly choose an augmentation policy each time to construct an intermediate augmentation, and lastly apply Cutout on the intermediate augmentation. Although this paper focuses on image classification tasks, which are the most natural choices for evaluating methods based on data augmentation techniques, data augmentations for other types of feature have also been studied recently. For example, one can use back-translation to augment text data, by which consistency training has been shown very effective for text classification (Edunov et al., 2018).

4. Experiments

In this section, we conduct experiments on various image datasets showing the effectiveness of our method compared with other state-of-the-art deep PLL methods.

4.1. Experiment Setup

Datasets. We employ five commonly used benchmark image datasets including Kuzushiji-MNIST (Clanuwat et al., 2018), Fashion-MNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011), CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009). We manually corrupt these datasets into partially

Table 1. Accuracy (mean±std) comparisons on Fashion-MNIST, Kuzushiji-MNIST, SVHN and CIFAR-10 with uniform partial labels on different ambiguity levels. The best result among each column is highlighted in bold.

Dataset	Method	$q = 0.1$	$q = 0.3$	$q = 0.5$	$q = 0.7$
Fashion-MNIST	Ours	$\mathbf{93.79 \pm 0.05\%}$	$\mathbf{93.72 \pm 0.20\%}$	$\mathbf{93.38 \pm 0.08\%}$	$\mathbf{92.19 \pm 0.03\%}$
	PiCO	$93.36 \pm 0.09\%$	$93.41 \pm 0.08\%$	$92.88 \pm 0.03\%$	$91.73 \pm 0.07\%$
	PRODEN	$89.15 \pm 0.58\%$	$89.10 \pm 0.26\%$	$88.22 \pm 0.35\%$	$85.87 \pm 0.28\%$
	VALEN	$89.30 \pm 0.72\%$	$89.06 \pm 0.42\%$	$88.23 \pm 0.29\%$	$86.05 \pm 0.25\%$
	LWS	$91.44 \pm 0.13\%$	$91.85 \pm 0.14\%$	$90.59 \pm 0.18\%$	$89.46 \pm 0.16\%$
	RC	$92.64 \pm 0.14\%$	$92.08 \pm 0.03\%$	$92.01 \pm 0.04\%$	$90.83 \pm 0.35\%$
	CC	$92.26 \pm 0.12\%$	$91.75 \pm 0.04\%$	$90.92 \pm 0.06\%$	$89.73 \pm 0.23\%$
	Fully Supervised	$93.92 \pm 0.07\%$			
Kuzushiji-MNIST	Ours	$\mathbf{98.27 \pm 0.07\%}$	$\mathbf{98.08 \pm 0.03\%}$	$\mathbf{97.44 \pm 0.04\%}$	$\mathbf{95.93 \pm 0.11\%}$
	PiCO	$97.68 \pm 0.06\%$	$97.34 \pm 0.07\%$	$97.15 \pm 0.03\%$	$91.90 \pm 0.04\%$
	PRODEN	$94.61 \pm 0.39\%$	$93.08 \pm 0.46\%$	$90.15 \pm 0.51\%$	$81.10 \pm 0.78\%$
	VALEN	$92.14 \pm 0.55\%$	$91.02 \pm 0.50\%$	$88.39 \pm 0.59\%$	$80.80 \pm 0.44\%$
	LWS	$96.22 \pm 0.10\%$	$96.15 \pm 0.24\%$	$95.43 \pm 0.02\%$	$93.63 \pm 0.04\%$
	RC	$96.84 \pm 0.09\%$	$96.31 \pm 0.15\%$	$96.17 \pm 0.06\%$	$95.84 \pm 0.12\%$
	CC	$96.45 \pm 0.04\%$	$96.16 \pm 0.02\%$	$95.62 \pm 0.10\%$	$95.33 \pm 0.14\%$
	Fully Supervised	$98.31 \pm 0.05\%$			
SVHN	Ours	$\mathbf{97.56 \pm 0.02\%}$	$\mathbf{97.19 \pm 0.06\%}$	$\mathbf{97.58 \pm 0.05\%}$	$\mathbf{95.46 \pm 0.23\%}$
	PiCO	$96.01 \pm 0.02\%$	$96.24 \pm 0.09\%$	$95.89 \pm 0.06\%$	$94.71 \pm 0.17\%$
	PRODEN	$96.35 \pm 0.25\%$	$96.07 \pm 0.23\%$	$95.61 \pm 0.21\%$	$94.35 \pm 0.29\%$
	VALEN	$94.51 \pm 0.38\%$	$93.90 \pm 0.20\%$	$93.14 \pm 0.42\%$	$92.21 \pm 0.49\%$
	LWS	$96.12 \pm 0.05\%$	$95.72 \pm 0.18\%$	$33.75 \pm 0.09\%$	$19.59 \pm 0.11\%$
	RC	$96.20 \pm 0.03\%$	$95.54 \pm 0.03\%$	$96.03 \pm 0.05\%$	$95.73 \pm 0.08\%$
	CC	$96.15 \pm 0.02\%$	$95.79 \pm 0.05\%$	$95.35 \pm 0.03\%$	$94.55 \pm 0.15\%$
	Fully Supervised	$97.58 \pm 0.03\%$			
CIFAR-10	Ours	$\mathbf{97.45 \pm 0.04\%}$	$\mathbf{97.28 \pm 0.02\%}$	$\mathbf{97.05 \pm 0.05\%}$	$\mathbf{95.77 \pm 0.08\%}$
	PiCO	$95.78 \pm 0.05\%$	$95.25 \pm 0.06\%$	$94.73 \pm 0.11\%$	$92.73 \pm 0.08\%$
	PRODEN	$91.94 \pm 0.32\%$	$91.10 \pm 0.50\%$	$89.82 \pm 0.47\%$	$86.48 \pm 0.47\%$
	VALEN	$84.87 \pm 0.43\%$	$82.95 \pm 0.72\%$	$80.63 \pm 0.89\%$	$72.59 \pm 0.79\%$
	LWS	$86.47 \pm 0.20\%$	$84.31 \pm 0.14\%$	$54.73 \pm 0.19\%$	$38.49 \pm 0.24\%$
	RC	$88.96 \pm 0.06\%$	$87.49 \pm 0.17\%$	$83.48 \pm 0.19\%$	$75.01 \pm 0.19\%$
	CC	$88.78 \pm 0.05\%$	$86.69 \pm 0.40\%$	$83.75 \pm 0.28\%$	$77.60 \pm 0.22\%$
	Fully Supervised	$97.57 \pm 0.03\%$			

labeled versions by uniform (Lv et al., 2020) and instance-dependent (Xu et al., 2021a) generating process.

To synthesize the uniform partial label datasets, we adopt the generating procedure used in (Lv et al., 2020). Specifically, we uniformly select each incorrect label $\bar{y}$ for each instance into its candidate label set with probability q. For the cases where none of the labels are flipped, we flip a random incorrect label into the candidate label set to guarantee all training examples are partially labeled.

Instance-dependent partial labels are generated by following the same generating procedure used in (Xu et al., 2021a). We corrupt Fashion-MNIST, Kuzushiji-MNIST and CIFAR-10 with the flipping probability depending on each specific instance. Specifically, given an instance $\boldsymbol{x}$, the flipping probability of each incorrect label is computed by $q_j(\boldsymbol{x}) = \frac{\hat{g}_j(\boldsymbol{x})}{\max_{k \in S} \hat{g}_k(\boldsymbol{x})}$, where $\hat{g}$ denotes a pre-trained neural network. In our experiments, we directly use the pre-trained network released by Xu et al. (2021a) to generate instance-dependent partial labels.

Compared methods. We compare our method against the following six deep PLL methods: (1) PiCO (Wang et al., 2022), a contrastive-learning-based method which identifies the true label using learned prototypes. (2) VALEN (Xu et al., 2021a), an instance-dependent PLL method which iteratively recovers the latent label distribution by employing the variational inference technique in training phase; (3) PRODEN (Lv et al., 2020), a self-training like method which progressively identifies the true labels using the output of the classier itself; (4) LWS (Wen et al., 2021), a discriminative PLL approach which considers the trade-off between losses in candidate labels and non-candidate labels; (5) RC (Feng et al., 2020), a risk-consistent PLL method which utilizes the importance re-weighting strategy; (6) CC

Table 2. Accuracy (mean±std) comparisons on CIFAR-100 with uniform partial labels on different ambiguity levels.

Dataset	Method	$q = 0.01$	$q = 0.05$	$q = 0.1$	$q = 0.2$
CIFAR-100	Ours	$\mathbf{83.12 \pm 0.20\%}$	$\mathbf{82.77 \pm 0.10\%}$	$\mathbf{82.24 \pm 0.07\%}$	$\mathbf{80.97 \pm 0.29\%}$
	PiCO	$74.39 \pm 0.15\%$	$73.97 \pm 0.09\%$	$51.94 \pm 0.11\%$	$20.29 \pm 0.04\%$
	PRODEN	$72.55 \pm 0.77\%$	$71.55 \pm 0.94\%$	$70.84 \pm 0.87\%$	$58.86 \pm 0.85\%$
	VALEN	$58.82 \pm 0.39\%$	$58.21 \pm 0.67\%$	$54.44 \pm 0.52\%$	$30.28 \pm 0.25\%$
	LWS	$58.54 \pm 0.12\%$	$55.19 \pm 0.23\%$	$40.12 \pm 0.34\%$	$23.90 \pm 0.18\%$
	RC	$64.95 \pm 0.23\%$	$62.48 \pm 0.14\%$	$57.48 \pm 0.04\%$	$44.13 \pm 0.23\%$
	CC	$63.74 \pm 0.17\%$	$61.22 \pm 0.21\%$	$58.65 \pm 0.06\%$	$51.65 \pm 0.49\%$
	Fully Supervised	$83.16 \pm 0.19\%$			

Table 3. Accuracy (mean±std) comparisons on Kuzushiji-MNIST, Fashion-MNIST, CIFAR-10 with instance-dependent partial labels.

Method	Kuzushiji-MNIST	Fashion-MNIST	CIFAR10
Ours	$\mathbf{95.07 \pm 0.06\%}$	$\mathbf{89.21 \pm 0.21\%}$	$87.80 \pm 0.11\%$
PiCO	$92.87 \pm 0.08\%$	$86.93 \pm 0.20\%$	$\mathbf{92.64 \pm 0.07\%}$
PRODEN	$87.71 \pm 0.62\%$	$84.25 \pm 0.61\%$	$76.51 \pm 0.69\%$
VALEN	$83.15 \pm 0.86\%$	$84.31 \pm 0.95\%$	$65.51 \pm 2.18\%$
LWS	$91.17 \pm 0.18\%$	$86.14 \pm 0.34\%$	$44.08 \pm 0.15\%$
RC	$94.00 \pm 0.12\%$	$89.10 \pm 0.24\%$	$75.90 \pm 0.34\%$
CC	$94.01 \pm 0.05\%$	$88.33 \pm 0.17\%$	$79.58 \pm 0.22\%$

Table 4. Accuracy (mean±std) of the degenerated method with uniform partial labels.

	$q = 0.5$	$q = 0.7$
Fashion-MNIST	$91.49 \pm 0.71\%$ $\downarrow \mathbf{(1.89\%)}$	$90.61 \pm 0.66\%$ $\downarrow \mathbf{(1.58\%)}$
Kuzushiji-MNIST	$95.99 \pm 0.06\%$ $\downarrow \mathbf{(1.45\%)}$	$92.28 \pm 0.20\%$ $\downarrow \mathbf{(3.65\%)}$
SVHN	$96.60 \pm 0.71\%$ $\downarrow \mathbf{(0.98\%)}$	$95.02 \pm 0.09\%$ $\downarrow \mathbf{(0.44\%)}$
CIFAR-10	$95.39 \pm 1.47\%$ $\downarrow \mathbf{(1.66\%)}$	$93.29 \pm 1.23\%$ $\downarrow \mathbf{(2.48\%)}$
	$q = 0.05$	$q = 0.1$
CIFAR-100	$76.07 \pm 0.15\%$ $\downarrow \mathbf{(6.70\%)}$	$74.93 \pm 0.19\%$ $\downarrow \mathbf{(7.31\%)}$

(Feng et al., 2020), a classifier-consistent PLL method which applies the cross entropy loss and transition matrix to form an empirical risk estimator.

To make fair comparisons, we basically use the same network architecture, learning rate, optimizer and augmentation strategy for all the comparing methods in the main experiment. There are particular cases in which the comparing methods with these default hyper-parameters fail to achieve comparable results, hence in these cases we use the suggested hyper-parameters specified in their original papers. The detailed description of the hyper-parameters is presented in the Appendix A.3.

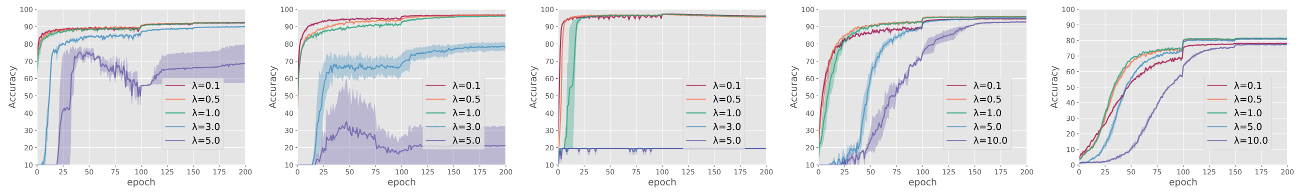
Implementation. Our implementation is based on PyTorch (Paszke et al., 2019), and experiments were carried out with NVIDIA Tesla V100 GPU. For Fashion-MNIST, Kuzushiji-MNIST, we employ LeNet-5 as the backbone neural network, while employ Wide-ResNet-34-10 for SVHN, CIFAR-10 and CIFAR-100. For all methods, we use SGD as the optimizer with a momentum of 0.9, a weight decay of 1e-4 and set batch size to 64. We set total epochs as 200, and the initial learning rate as 0.1 while divided it by 10 after 100 and 150 epochs respectively. For our method, we simply set $T' = 100$, $\lambda = 1$ and $K = 3$ across all datasets. We present the mean and standard deviation in each case based on three independent runs with different random seeds.

4.2. Experiment Results

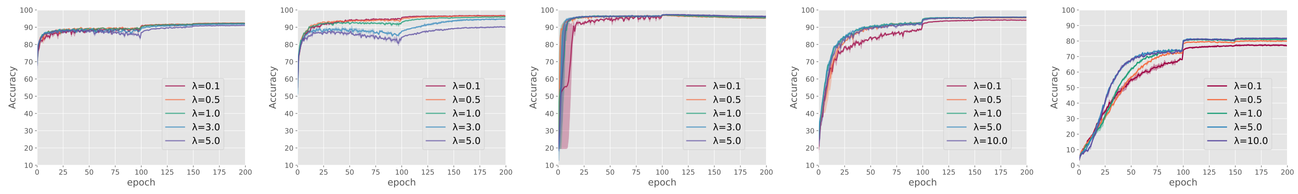
Table 1 reports the comparison results on Fashion-MNIST, Kuzushiji-MNIST, SVHN and CIFAR-10. As is shown, our method consistently outperforms all comparing methods, and the improvements are particularly noticeable on high ambiguity levels. For example, on CIFAR-10, our method improves upon the current state-of-the-art method PiCO by $\mathbf{1.67}\%$, $\mathbf{2.03}\%$, $\mathbf{2.32}\%$ and $\mathbf{3.04}\%$ on four different ambiguity levels respectively. Our method achieves comparable accuracy even compared with fully supervised learning. Particularly, with $q = 0.1, 0.2$ and 0.5, our method gives very small performance drop (lower than 1%) compared with fully supervised learning on all these four datasets. Moreover, even with $q = 0.7$, the performance drop of our method is lower than 3% compared with fully supervised learning. On the contrary, comparing methods like LWS and VALEN work well on low ambiguity levels, but drop dramatically as ambiguity increases.

Table 2 reports the comparing results on CIFAR-100. As is shown, the superiority of our method is more significant compared with that on previous simpler datasets. Even with the presence of near 20 false-positive labels for each instance, our method is still competitive compared with fully supervised learning (with only 2.19% performance drop).

In addition to the uniform partial label setting, we also conducted experiments with instance-dependent partial labels on Kuzushiji-MNIST, Fashion-MNIST and CIFAR-10. Table 3 shows that our method outperforms all baselines on Kuzushiji-MNIST and Fashion-MNIST. Although the accuracy of our method is lower than that of PiCO on CIFAR-10, it is still acceptable and significantly higher than the rest of comparing methods.



(a) Fixed λ Fashion-MNIST. (b) Fixed λ on Kuzushiji-MNIST. (c) Fixed λ on SVHN. (d) Fixed λ on CIFAR-10. (e) Fixed λ on CIFAR-100.



(f) Dynamic λ Fashion-MNIST. (g) Dynamic λ on Kuzushiji-MNIST. (h) Dynamic λ on SVHN. (i) Dynamic λ on CIFAR-10. (j) Dynamic λ on CIFAR-100.

Figure 2. Accuracy curves with different balancing strategies. The top row shows results with fixed λ and the bottom row shows the results with dynamic λ. Dark colors show the mean accuracy of 5 trials and light colors show standard deviation.

4.3. Ablation Study

In this subsection, we conduct extensive experiments to verify the importance of different components of our method.

Analysis of the balancing factor. We study the effect of the balancing factor by comparing performance of our method with $\lambda \in \{0.1, 0.5, 1.0, 3.0, 5.0, 10.0\}$ in both fixed and dynamic strategies. This ablation experiment is conducted on the first four datasets with $q = 0.7$ and CIFAR-100 with $q = 0.2$, where the implementation details are same with those in the main experiment. As is shown in Figure 2: (1) Performance under fixed factor exhibits large fluctuations in the early stage, especially when large λ is employed; On the contrary, dynamic strategy gives stable performance during the entire training phase. (2) Using fixed factor fails in some cases, while dynamic strategy achieves reasonable performance in all cases. (3) When using dynamic strategy, a relative larger λ always gives good performance.

The importance of regularization term. As regularization plays an important role in our method, we next explore the influence of the proposed consistency regularization term. Specifically, we replace the conformal label distribution inferrd by Eq. (7) with simply re-normalized model outputs, and remaining the augmentation techniques. This degenerated method could be considered as the combination of training with loss (2) and the augmented self-training. The comparing results on different datasets and ambiguity levels are shown in Table 4, where the values in bold indicate the accuracy gap compared to our method. We can see that the performance declines more obviously in highly ambiguous cases. The accuracy gap is particularly large on CIFAR-100, which reflects the superiority of our consistency regularization method on difficult learning task.

Table 5. Results with and without the supervised loss.

	with supervised loss	without supervised loss
F-MNIST (0.7)	$\mathbf{92.19 \pm 0.03}$%	91.78 ± 0.09%
K-MNIST (0.7)	$\mathbf{95.93 \pm 0.11}$%	94.24 ± 0.09%
SVHN (0.7)	$\mathbf{95.46 \pm 0.23}$%	70.92 ± 36.34%
CIFAR-10 (0.7)	$\mathbf{95.77 \pm 0.08}$%	92.80 ± 0.14%
CIFAR-100 (0.1)	$\mathbf{82.24 \pm 0.07}$%	80.65 ± 0.04%

The important of supervised loss. As we discussed, supervised loss on non-candidate labels also plays an important role in our framework. To examine the impact of supervised loss, We compare our method with a degenerate method that only uses regularization loss to learn the model. The results in Table 5 show that without supervised loss, simply training with consistency loss degrades the performance and even fails in the end of training on SVHN. Similar results are reported in the experiment of the balancing factor (see Figure 2), in which large balancing factors caused significant performance drops in some cases.

The influence of backbone network. With more powerful backbone network, the comparing methods achieve better performance in most cases compared with the results reported in their original papers. However, there are particular cases that our re-implemented results are lower than the original implementations, e.g. PiCO on CIFAR-100. Table 6 shows the comparison of PiCO and our method with different backbones, where ResNet-18 is that used in the PiCO literature. We report the accuracy decrease of each method on $p = 0.01/0.05/0.10$ compared with learning from full supervision. As is shown, our method yields lower performance drop compared with PiCO, when increasing the ambiguity degree. Especially, with $p = 0.10$, the performance

Table 6. Accuracy (mean±std) comparison between PiCO and Our method with different backbones. WRN-34-10 is short for Wide-ResNet-34-10.

		PiCO	Ours
ResNet-18	Fully Supervised	$73.56 \pm 0.10\%$	$79.88 \pm 0.03\%$
	$p = 0.01$	$73.09 \pm 0.34\%$ $\downarrow$ **(0.47%)**	$79.54 \pm 0.12\%$ $\downarrow$ **(0.34%)**
	$p = 0.05$	$72.74 \pm 0.30\%$ $\downarrow$ **(0.82%)**	$78.96 \pm 0.06\%$ $\downarrow$ **(0.92%)**
	$p = 0.10$	$69.91 \pm 0.24\%$ $\downarrow$ **(3.65%)**	$77.72 \pm 0.08\%$ $\downarrow$ **(2.15%)**
WRN-34-10	Fully Supervised	$74.47 \pm 0.14\%$	$83.16 \pm 0.19\%$
	$p = 0.01$	$74.39 \pm 0.15\%$ $\downarrow$ **(0.08%)**	$83.12 \pm 0.20\%$ $\downarrow$ **(0.04%)**
	$p = 0.05$	$73.97 \pm 0.09\%$ $\downarrow$ **(0.50%)**	$82.77 \pm 0.10\%$ $\downarrow$ **(0.39%)**
	$p = 0.10$	$51.94 \pm 0.11\%$ $\downarrow$ **(22.53%)**	$82W.24 \pm 0.07\%$ $\downarrow$ **(0.92%)**

of PiCO drops dramatically with Wide-ResNet-34-10, while our method is robust for different backbone networks.

The influence of K (the number of augmented images). As stated in Section 4.2, our proposed method performs well with K = 3. Now we investigate how the performance varies with different K. We also compare the differences between optimizing the conformal label distribution and the KL divergence of all pairs $\mathcal{A}(\boldsymbol{x})$. As shown in Table 7, even with small K, the performance of our method is quite good and larger K does not achieve very obvious improvement. Moreover, the performance of optimizing pairwise KL divergence is worse than that of optimizing conformal distribution.

The influence of data augmentation. In the above experiments, the consistency loss is calculated using three augmentations ($K = 3$) including one weak and two strong augmentations. Now we compare the performance of different augmentation strategies. As we can see in Table 8 (e.g., 1S+2W means one strong and two weak augmentations are used simultaneously), using only strong augmentations or the combinations of strong and weak augmentations achieves good performance, while only using weak augmentations results in worse performance.

5. Conclusion

In this work, we revisited the utilization of consistency regularization in PLL literature and for the first time proposed a regularized training framework for deep PLL. We presented an effective regularization term by involving a conformal label distribution for each instance, which could be adaptively inferred during training in the bi-level optimization framework. Experimental results on both uniform and instance-dependent partial label datasets demonstrated the effectiveness of our concise method compared with the current state-of-the-art deep PLL methods. Moreover, our method achieved competitive performance even compared with fully supervised learning.

Table 7. Results (%) with different choices of K on Fashion-MNIST, Kuzushiji-MNIST, CIFAR-10 and CIFAR-100.

		K=2	K=3	K=4	K=5
F-MNIST (ID)	Pairwise	88.01±0.03	87.23±0.16	85.80±0.16	53.89±31.25
	Conformal	**89.60±0.01**	**89.44±0.10**	**89.54±0.10**	**89.45±0.07**
K-MNIST (ID)	Pairwise	94.01±0.15	91.23±0.13	82.23±0.64	51.48±29.37
	Conformal	**95.44±0.21**	**95.15±0.13**	**95.29±0.16**	**95.30±0.08**
CIFAR-10 (0.7)	Pairwise	94.65±0.10	94.99±0.10	95.28±0.07	95.29±0.03
	Conformal	**95.44±0.12**	**95.77±0.08**	**95.89±0.03**	**95.83±0.10**
CIFAR-10 (ID)	Pairwise	81.01±4.28	83.20±4.50	83.26±4.66	86.62±0.09
	Conformal	**89.46±1.09**	**87.80±0.11**	**87.53±0.19**	**87.90±0.11**
CIFAR-100 (0.1)	Pairwise	81.59±0.20	82.17±0.03	**82.41±0.12**	**82.58±0.30**
	Conformal	**81.94±0.21**	**82.24±0.07**	82.21±0.12	82.20±0.07

Table 8. Result (%) with different augmentations on CIFAR-10. ▼ indicates the worst case of each row.

	3×S	1×S + 2×W	2×S + 1×W	3×W
CIFAR-10 (0.7)	95.74±0.17	95.73±0.10	95.77±0.08	93.74±0.12▼
CIFAR-10 (ID)	90.22±0.22	88.78±0.62	87.80±0.11	87.45±0.57▼
CIFAR-100 (0.1)	82.35±0.36	82.37±0.13	82.24±0.07	78.43±0.03▼

References

Chen, C.-H., Patel, V. M., and Chellappa, R. Learning from ambiguously labeled face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(7):1653–1667, 2017.

Chou, Y.-T., Niu, G., Lin, H.-T., and Sugiyama, M. Unbiased risk estimators can mislead: A case study of learning with complementary labels. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 1929–1938, 2020.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Cour, T., Sapp, B., and Taskar, B. Learning from partial labels. *The Journal of Machine Learning Research*, 12: 1501–1536, 2011.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the 32th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

DeVries, T. and Taylor, G. W. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Edunov, S., Ott, M., Auli, M., and Grangier, D. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

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

Feng, L. and An, B. Partial label learning with self-guided retraining. In *Proceedings of the 35th Association for the Advancement of Artificial Intelligence*, volume 33, pp. 3542–3549, 2019.

Feng, L., Lv, J., Han, B., Xu, M., Niu, G., Geng, X., An, B., and Sugiyama, M. Provably consistent partial-label learning. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, pp. 6575–6582, 2020.

Gao, Y. and Zhang, M.-L. Discriminative complementary-label learning with weighted loss. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 3587–3597, 2021.

Gong, C., Liu, T., Tang, Y., Yang, J., Yang, J., and Tao, D. A regularization approach for instance-based superset label learning. *IEEE Transactions on Cybernetics*, 48(3): 967–978, 2017.

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

Jeroslow, R. G. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32(2):146–164, 1985.

Jin, R. and Ghahramani, Z. Learning with multiple labels. In *Proceedings of the 16th Annual Conference on Neural Information Processing Systems*, pp. 897–904, 2002.

Kiryo, R., Niu, G., Plessis, M. C. d., and Sugiyama, M. Positive-unlabeled learning with non-negative risk estimator. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems*, pp. 1675–1685, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. *Technique Report*, 2009.

Liu, L. and Dietterich, T. G. A conditional multinomial mixture model for superset label learning. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems*, pp. 557–565, 2012.

Liu, W., Shen, X., Wang, H., and Tsang, I. W. The emerging trends of multi-label learning. *arXiv preprint arXiv:2011.11197*, 2020.

Luo, J. and Orabona, F. Learning from candidate labeling sets. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems*, pp. 1504–1512, 2010.

Lv, J., Xu, M., Feng, L., Niu, G., Geng, X., and Sugiyama, M. Progressive identification of true labels for partial-label learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 6500–6510, 2020.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. Learning with noisy labels. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, pp. 1196–1204, 2013.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

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

Ni, P., Zhao, S.-Y., Dai, Z.-G., Chen, H., and Li, C.-P. Partial label learning via conditional-label-aware disambiguation. *Journal of Computer Science and Technology*, 36(3):590–605, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33th Annual Conference on Neural Information Processing Systems*, pp. 8024–8035, 2019.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, volume 33, 2020.

Su, G., Chen, W., and Xu, M. Positive-unlabeled learning from imbalanced data. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pp. 2995–3001, 2021.

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

Wang, H., Xiao, R., Li, Y., Feng, L., Niu, G., Chen, G., and Zhao, J. Contrastive label disambiguation for partial label learning. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

Wang, Y., Han, J., Shen, Y., and Hui, X. Pointwise manifold regularization for semi-supervised learning. *Frontiers of Computer Science*, 15(151303), 2021.

Wen, H., Cui, J., Hang, H., Liu, J., Wang, Y., and Lin, Z. Leveraged weighted loss for partial label learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 11091–11100, 2021.

Wu, J.-H., Wu, X., Chen, Q.-G., Hu, Y., and Zhang, M.-L. Feature-induced manifold disambiguation for multi-view partial multi-label learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 557–565, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. Unsupervised data augmentation for consistency training. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, volume 33, 2020.

Xu, M. and Guo, L.-Z. Learning from group supervision: the impact of supervision deficiency on multi-label learning. *Science China Information Sciences*, 64(3):1–13, 2021.

Xu, N., Qiao, C., Geng, X., and Zhang, M.-L. Instance-dependent partial label learning. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, volume 34, 2021a.

Xu, Y., Shang, L., Ye, J., Qian, Q., Li, Y.-F., Sun, B., Li, H., and Jin, R. Dash: Semi-supervised learning with dynamic thresholding. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 11525–11536, 2021b.

Yan, Y. and Guo, Y. Multi-level generative models for partial label learning with non-random label noise. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pp. 3264–3270, 2021.

Yao, Y., Deng, J., Chen, X., Gong, C., Wu, J., and Yang, J. Deep discriminative cnn with temporal ensembling for ambiguously-labeled image classification. In *Proceedings of the 36th Association for the Advancement of Artificial Intelligence*, volume 34, pp. 12669–12676, 2020a.

Yao, Y., Gong, C., Deng, J., and Yang, J. Network cooperation with progressive disambiguation for partial label learning. In *Proceedings of the 13th Machine Learning and Knowledge Discovery in Databases*, volume 12458, pp. 471–488, 2020b.

Yu, F. and Zhang, M.-L. Maximum margin partial label learning. In *Proceedings of the 7th Asian Conference on Machine Learning*, volume 45, pp. 96–111, 2016.

Zeng, Z., Xiao, S., Jia, K., Chan, T.-H., Gao, S., Xu, D., and Ma, Y. Learning by associating ambiguously labeled images. In *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 708–715, 2013.

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

Zhang, M.-L. and Zhou, Z.-H. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2013.

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

Zhu, X. and Goldberg, A. B. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 3(1):1–130, 2009.

A. Appendix

A.1. Proof of Proposition 1

Proposition 1. *The loss function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p})$ is convex in the distribution $\boldsymbol{p}$, i.e.,*

$$\mathcal{L}(\boldsymbol{\theta}, \alpha\boldsymbol{p}_1 + (1-\alpha)\boldsymbol{p}_2) \leq \alpha\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_1) + (1-\alpha)\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_2),$$

where $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are two distributions and $0 \leq \alpha \leq 1$.

Proof. For this proof, we use an inequality called the *log-sum inequality*, which states that for non-negative real numbers a_1, a_2, b_1, b_2, we have

$$(a_1 + a_2) \cdot \log\left(\frac{a_1 + a_2}{b_1 + b_2}\right) \leq a_1 \cdot \log\left(\frac{a_1}{b_1}\right) + a_2 \cdot \log\left(\frac{a_2}{b_2}\right).$$

For training example $(\boldsymbol{x}, \mathcal{S})$, we have

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{\theta}, \alpha\boldsymbol{p}_1 + (1-\alpha)\boldsymbol{p}_2) \\
&\quad= \mathcal{L}_{sup} + \textstyle\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \mathtt{KL}\left(\alpha\boldsymbol{p}_1 + (1-\alpha)\boldsymbol{p}_2 || g(\boldsymbol{z})\right) \\
&\quad= \mathcal{L}_{sup} + \textstyle\sum_{k\in\mathcal{S}}\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \left(\alpha p_{1,k} + (1-\alpha)p_{2,k}) \cdot \log\left(\frac{\alpha p_{1,k} + (1-\alpha)p_{2,k}}{\alpha g_k(\boldsymbol{z}) + (1-\alpha)g_k(\boldsymbol{z})}\right)\right) \\
&\quad\leq \mathcal{L}_{sup} + \textstyle\sum_{k\in\mathcal{S}}\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \left(\alpha p_{1,k} \cdot \log\left(\frac{\alpha p_{1,k}}{\alpha g_k(\boldsymbol{z})}\right) + (1-\alpha)p_{2,k} \cdot \log\left(\frac{(1-\alpha)p_{2,k}}{(1-\alpha)g_k(\boldsymbol{z})}\right)\right) \\
&\quad= \alpha\mathcal{L}_{sup} + \alpha\textstyle\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \mathtt{KL}(\boldsymbol{p}_1 || g(\boldsymbol{z})) + (1-\alpha)\mathcal{L}_{sup} + (1-\alpha)\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} \mathtt{KL}(\boldsymbol{p}_2 || g(\boldsymbol{z})) \\
&\quad= \alpha\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_1) + (1-\alpha)\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}_2).
\end{aligned}$$

□

A.2. Derivation of Eq. (7)

Based on the proposition that the loss function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p})$ is convex w.r.t. $\boldsymbol{p}$, we derive the conformal label distribution $\boldsymbol{p}$ for an specific instance $(\boldsymbol{x}, \mathcal{S})$ by using Lagrangian Multiplier method. The elements of $\boldsymbol{p}$ on non-candidate labels are simply revealed in the constraint $p_k = 0, \forall k \notin \mathcal{S}$, thus we only focus on the elements of $\boldsymbol{p}$ on candidate labels. We transform the original inner problem in Eq. (5) into the minimization of the corresponding Lagrangian function:

$$\underset{p_k \forall k\in\mathcal{S}, \mu}{\arg\min} \left(\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}) + \mu\left(\textstyle\sum_{k\in\mathcal{S}} p_k - 1\right)\right),$$

where μ is the Lagrange multiplier to be optimized. The optimum solution of this problem needs to satisfy the following KKT conditions:

$$\frac{\partial\left[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{p}) + \mu\left(\sum_{k\in\mathcal{S}} p_k - 1\right)\right]}{\partial p_k} = 0,\ \forall k \in \mathcal{S}; \quad \textstyle\sum_{k\in\mathcal{S}} p_k - 1 = 0; \quad \mu \geq 0.$$

Based on the first equation, we have

$$\begin{aligned}
&\textstyle\sum_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} (\log p_k - \log g_k(\boldsymbol{z}) + 1) + \mu = 0,\ k \in \mathcal{S}, \\
\implies\ & \log p_k + \left(1 + \frac{\mu}{|\mathcal{A}(\boldsymbol{x})|}\right) = \frac{1}{|\mathcal{A}(\boldsymbol{x})|} \log\left(\textstyle\prod_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} g_k(\boldsymbol{z})\right),\ k \in \mathcal{S}, \\
\implies\ & p_k \cdot \exp\left(1 + \frac{\mu}{|\mathcal{A}(\boldsymbol{x})|}\right) = \left(\textstyle\prod_{\boldsymbol{z}\in\mathcal{A}(\boldsymbol{x})} g_k(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}},\ k \in \mathcal{S}.
\end{aligned}$$

Let $\eta = \exp\left(1 + \frac{\mu}{|\mathcal{A}(\boldsymbol{x})|}\right)$, then by combining the above derivation and the second KKT conditeion, we have

$$\eta = \sum_{k \in \mathcal{S}} \left(\prod_{\boldsymbol{z} \in \mathcal{A}(\boldsymbol{x})} g_k(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}}.$$

Therefore, for each $k \in \mathcal{S}$, the corresponding optimal p_k^* can be calculated as

$$p_k^* = \frac{\left(\prod_{\boldsymbol{z} \in \mathcal{A}(\boldsymbol{x})} g_k(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}}}{\sum_{j \in \mathcal{S}} \left(\prod_{\boldsymbol{z} \in \mathcal{A}(\boldsymbol{x})} g_j(\boldsymbol{z})\right)^{\frac{1}{|\mathcal{A}(\boldsymbol{x})|}}}.$$

A.3. Experiment Details

A.3.1. Datasets

We use four widely-used benchmark datasets, i.e., Kuzushiji-MNIST, Fasihon-MNSIT, SVHN, CIFAR-10, CIFAR100. We describe these five datasets as follows:

- **Fashion-MNIST** consists of 70,000 28×28 grey scale images. It has 10 fashion items as the classes including T-shirt/top, trouser, pullover, dress, sandal, coat, shirt, sneaker, bag, and ankle boot.
- **Kuzushiji-MNIST** consists of 70,000 28×28 grey scale images and each image is associated with one label of 10-class cursive Japanese (”Kuzushiji”) characters.
- **SVHN** consists of 73257 $32 \times 32 \times 3$ colored images in RGB format. It has 10 classes from 0 to 9, which denote the house numbers from Google Street View images.
- **CIFAR-10** consists of 60,000 $32 \times 32 \times 3$ colored images in RGB format. It has 10 classes including airplane, bird, automobile, cat, deer,frog, dog, horse, ship, and truck.
- **CIFAR-100** is just like the CIFAR10, consisting of 60,000 $32 \times 32 \times 3$ colored images in RGB format. It has totally 100 classes and each class contains 600 images. These 100 classes are grouped into 20 superclasses. Each image comes with a“fine” label and a “coarse” label.

The data augmentation techniques used for Fashion-MNIST, Kuzushiji-MNIST are: (1) Random Horizontal Flipping, (2) Random Cropping, and (2) Cutout. The data augmentation techniques used for CIFAR-10, CIFAR-100 and SVHN are: (1) Random Horizontal Flipping, (2) Random Cropping, (3) Cutout, and (4) AutoAugment.

A.3.2. Implementations of Comparing Methods

The detailed configurations is shown in Table 9 with the following descriptions:

1) The data augmentation strategy in used for PiCO is different from our method. In Table 9, Augmentation* denotes the data augmentation strategy suggested by the PiCO literature. We also remain the total epochs (800) suggested in their paper.
2) LWS and VALEN perform poorly on CIFAR-100 when using our default configurations, so we remain the configurations suggested in their released code package.
3) Since VALEN fails when initial learning rate is set to 1e-1, we set 1e-2 as the initial learning rate on all datasets except for CIFAR-100.
4) Besides the configurations illustrated in Table 9, we remain the hyper-parameters which are specifically used in each comparing method according to their released code package.

Table 9. Implementation details of the comparing methods

	Uniform partial labels		Instance-depedent partial labels
	Fashion-MNIST Kuzushiji-MNIST SVHN CIFAR-10	CIFAR-100	Fashion-MNIST Kuzushiji-MNIST CIFAR-10
Ours	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200
VALEN	WideResNet Augmentation Initial Learning Rate 1e-2 Weight Decay 1e-4 Epochs 200	ResNet No Augmentation Initial Learning Rate 5e-2 Weight Decay 1e-3 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-2 Weight Decay 1e-4 Epochs 200
PiCO	WideResNet Augmentation* Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 800	WideResNet Augmentation* Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 800	WideResNet Augmentation* Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 800
LWS	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	ConvNet No Augmentation Initial Learning Rate 1e-2 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200
PRODEN	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200
RC	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200
CC	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200	WideResNet Augmentation Initial Learning Rate 1e-1 Weight Decay 1e-4 Epochs 200