SoLar: Sinkhorn Label Refinery for Imbalanced Partial-Label Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

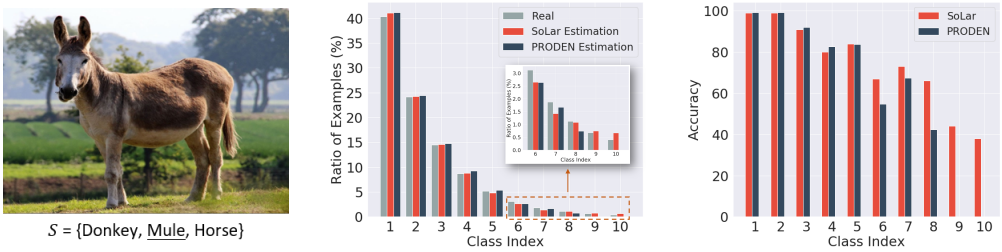
Partial-label learning (PLL) is a peculiar weakly-supervised learning task where the training samples are generally associated with a set of candidate labels instead of single ground truth. While a variety of label disambiguation methods have been proposed in this domain, they normally assume a class-balanced scenario that may not hold in many real-world applications. Empirically, we observe degenerated performance of the prior methods when facing the combinatorial challenge from the long-tailed distribution and partial-labeling. In this work, we first identify the major reasons that the prior work failed. We subsequently propose **SoLar**, a novel Optimal Transport-based framework that allows to refine the disambiguated labels towards matching the marginal class prior distribution. SoLar additionally incorporates a new and systematic mechanism for estimating the long-tailed class prior distribution under the PLL setup. Through extensive experiments, SoLar exhibits substantially superior results on standardized benchmarks compared to the previous state-of-the-art PLL methods.

1 Introduction

The remarkable success of deep learning typically requires collecting massive labeled data, which is notoriously labor-intensive. Of particular concern, data labeling can suffer from inherent and pervasive label ambiguity. Take Figure 1 (a) as an example, a Mule can be visually similar to both Donkeys and Horses, which hinders non-expert annotators from recognizing the true breed. Similar cases also arise in many real-world applications, such as automatic image annotation [1], bird song recognition [2], and facial age estimation [3]. To deal with ambiguous supervision, *partial-label learning* (PLL) [4, 5], which allows each training example to be annotated by a set of candidate labels, has attracted significant attention from the community. A plethora of methods have been developed to tackle this problem, including pseudo-labeling methods [6, 7], graph-based methods [8, 9, 10], etc.

Despite the promise, existing PLL methods have been commonly driven by the assumption that training data consists of class-balanced distribution, which may not hold in practice. In many real-world scenarios, training data exhibits a *long-tailed label distribution* [11]. That is, many labels occur infrequently in the training data. Concerningly, the imbalanced data can exacerbate the fundamental challenge of label disambiguation, *i.e.*, identifying the ground-truth label from candidates. Indeed, we find that current best-performing PLL methods display degenerated performance in the long-tailed setting. This happens because the predictions of pseudo-labeling—a core component that PLL methods rely on—can be largely biased towards the head and majority classes. We exemplify the phenomenon in Figure 1 (b), where a strong method PRODEN [6] rarely assigns probability mass to the tail classes (indexed by 9 and 10). As a result, these tail classes suffer from almost zero accuracy, as shown in Figure 1 (c). To date, few efforts have been made to resolve this.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) Example of label ambiguity (b) Class prior estimation (c) Class-wise accuracy

Figure 1: (a) An input image with three candidate labels, where the ground-truth is `Mule`. (b) The real/estimated class distribution (by counting real/predicted labels) on a long-tailed version of CIFAR10 dataset with partial labels (more details in Section 4). The class prior estimated by SoLar is very close to the real one, while PRODEN fails to recognize some tail labels. (c) Class-wise performance comparison of PRODEN and SoLar on the same dataset, where SoLar performs much better on data-scarce classes.

Motivated by this, we propose a novel framework for long-tailed partial-label learning, called **S**inkh**o**rn **La**bel **r**efinery (dubbed **SoLar**). Our framework emphasizes the long-tailed nature of training data, while performing label disambiguation for partial-label learning. Our key idea is to enforce constraints on pseudo-labels—the distribution of which should *match the class prior distribution*. The constraints can incentivize the label disambiguation process to select tail labels from candidate sets, instead of always selecting head classes. We formalize the idea as an optimal transport problem [12] that searches for a proper label assignment subjected to constraints: (1) the probability mass is distributed within candidate sets; (2) the distribution of (predicted) pseudo-labels matches the prior class distribution. The first constraint is natural in the PLL setup, and the latter one uniquely empowers SoLar to disambiguate tail labels from candidates. We show that our constrained optimization objective can be tractably solved using the Sinkhorn-Knopp algorithm [13], which incurs only a few computational overheads. Theoretically, we prove that our overall objective is statistically consistent, which ensures the optimality of the learned classifier at a population level.

As an integral part of our framework, SoLar tackles the challenge of estimating the class prior distribution, which is necessitated in our constrained optimization. Unlike conventional long-tailed learning (LTL) scenarios [14, 15], the PLL setup is presented with label ambiguity in the proximity of candidate sets. This makes it *intractable* to estimate the marginal class prior distribution, especially by counting training samples grouped by class labels. We tackle this non-trivial problem by performing an iterative class prior estimation, which provides a strong proxy for real prior (Section 3.2). Empirically in Figure 1 (b), we show that the estimated class prior favorably matches the ground-truth prior on a long-tailed version of the CIFAR10 dataset with partial labels.

We comprehensively evaluate SoLar on various benchmark datasets, where SoLar establishes state-of-the-art performance. Compared to the best baseline, SoLar improves the accuracy on tail classes by **15.12**% on the long-tailed version of the CIFAR10 dataset. While our work primarily focuses on label disambiguation from the PLL perspective, we demonstrate that SoLar can be compatible with LTL techniques as well. For example, we equip SoLar with logit adjustment [14], a state-of-the-art LTL method and further improve SoLar's performance by **6.72**% on long-tailed CIFAR10. We hope our work will inspire future works to tackle this important problem.

2 Problem Setup

Let $\mathcal{X} = \mathbb{R}^d$ denote the input space and $\mathcal{Y} = \{1, \dots, L\}$ denote the label space. We receive a training dataset $\mathcal{D} = \{(\boldsymbol{x}_i, S_i)\}_{i=1}^n$ with n examples. Each tuple in $\mathcal{D}$ consists of an image $\boldsymbol{x}_i \in \mathcal{X}$ and the candidate label set $S_i \subset \mathcal{Y}$. Following previous works [6, 7], we assume the true label $y_i \in \mathcal{Y}$ of $\boldsymbol{x}_i$ is concealed in S_i, *i.e.*, $y_i \in S_i$. A fundamental challenge in partial-label learning is label disambiguation, *i.e.*, identifying the ground-truth label y_i from the candidate label set S_i.

Our goal is to train a classifier $f : \mathcal{X} \mapsto [0, 1]^L$, parameterized by θ, that can perform predictions on unseen testing data. Here f is the softmax output of a neural network, and $f_j(\cdot)$ denotes the j-th entry. Besides, we denote the prediction matrix by $\boldsymbol{P} = [\boldsymbol{p}_1, \dots, \boldsymbol{p}_n]^\top = [p_{ij}]_{n \times L}$, where $p_{ij} = f_j(\boldsymbol{x}_i)$. To perform label disambiguation, we maintain a pseudo-label matrix $\boldsymbol{Q} = [\boldsymbol{q}_1, \dots, \boldsymbol{q}_n]^\top \in [0, 1]^{n \times L}$ and train the classifier with the cross-entropy loss $l_{\text{ce}}(f; \boldsymbol{x}_i, \boldsymbol{q}_i) = \sum_{j=1}^L -q_{ij} \log(f_j(\boldsymbol{x}_i))$, where q_{ij} is the j-th entry of $\boldsymbol{q}_i$.

Algorithm 1: Pseudo-code of SoLar.

Input: Training dataset $\mathcal{D}$, classifier f, uniform marginal $\boldsymbol{r}, \boldsymbol{c}$, hyperparameters ρ, τ, μ, λ.
for $epoch = 1, 2, \ldots,$ **do**
  for $step = 1, 2, \ldots,$ **do**
    Get classifier prediction $\boldsymbol{P}$ on a mini-batch of data B of size n_B
    Calculate $\boldsymbol{M}$ such that $m_{ij} = p_{ij}^{\lambda} \mathbb{I}(j \in S_i)$
    for $t = 1, \ldots, T$ **do**
      $\boldsymbol{\alpha} \leftarrow \boldsymbol{c}./(\boldsymbol{M}\boldsymbol{\beta}), \quad \boldsymbol{\beta} \leftarrow \boldsymbol{r}./(\boldsymbol{M}^{\top}\boldsymbol{\alpha})$ `// Sinkhorn-Knopp iteration`
    end
    $\boldsymbol{Q} = n_B \cdot \mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{M}\mathrm{diag}(\boldsymbol{\beta})$ `// Compute pseudo-labels`
    for $j = 1, \ldots, L$ **do**
      Generate a subset $B_j = \{(\boldsymbol{x}_i, \boldsymbol{q}_i) | j = \arg\max_{j' \in S_i} q_{ij'}\}$ from B
      Select reliable samples by small loss l_i and high-confidence $e_i > \tau$
      `// Class-wise reliable sample selection`
    end
    Train the classifier f by minimizing l_{cls}
  end
  Compute $\boldsymbol{z}$ by $z_j = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(j = \arg\max_{j' \in S_i} f_{j'}(\boldsymbol{x}_i))$ `// Empirical class prior estimation`
  $\boldsymbol{r} \leftarrow \mu\boldsymbol{r} + (1-\mu)\boldsymbol{z}$ `// Moving-average distribution updating`
end

3 Proposed Method

In this section, we describe our novel **S**inkh**o**rn **La**bel **r**efinery (SoLar) framework for partial-label learning. As we reckon, the combinatorial challenge of partial-labeling and long-tail learning lies in matching between a decent marginal prior distribution with drawing the pseudo labels. To cope with these problems, SoLar comprises two components. On one hand, SoLar formulates an optimal transport objective to facilitate the refinement of pseudo-labels to match the marginal class prior. The resulted constrained optimization objective encourages the model to properly draw the tail labels. On the other hand, the PLL setting manifests difficulty in straightforwardly estimating the class prior. SoLar henceforth incorporates two novel techniques—a moving-average estimation approach with a complementary reliable set sampling mechanism. *The pseudo-code is summarized in Algorithm 1.*

3.1 Sinkhorn Pseudo-Label Refinery

Optimal Transport Objective. We formalize an optimal transport problem for proper label assignments. At each training step, SoLar searches for pseudo-labels $\boldsymbol{Q}$ close to the current classifier's predictions $\boldsymbol{P}$, while subject to some constraints:

$$\begin{aligned} \min_{\boldsymbol{Q} \in \Delta} E(\boldsymbol{Q}, \boldsymbol{P}) &= \langle \boldsymbol{Q}, -\log(\boldsymbol{P}) \rangle = -\sum_{i=1}^{n} \sum_{j=1}^{L} q_{ij} \log p_{ij} \\ \text{s.t. } \Delta &= \{[q_{ij}]_{n \times L} | \boldsymbol{Q}^{\top} \mathbf{1}_n = \boldsymbol{r}, \boldsymbol{Q} \mathbf{1}_L = \boldsymbol{c}, q_{ij} = 0 \text{ if } j \notin S_i\} \subset [0,1]^{n \times L}. \end{aligned} \tag{1}$$

$\boldsymbol{r}$ is an L-dimensional probability simplex that indicates the prior class distribution. In other words, we would like the summation of the j-th column in $\boldsymbol{Q}$ (*i.e.*, the total probability mass predicted as class j) to match the prior class probability. Note that, here we temporarily assume we have a decent estimation of the class priors and we will describe the means of estimation in the following Section 3.2. The column vector $\boldsymbol{c} = \frac{1}{n}\mathbf{1}_n$ indicates that our n training examples are sampled uniformly. In Eq. (1), with a slight abuse of notation, $\boldsymbol{Q}$ is essentially a joint probability matrix post to the refinery. That is, each $\boldsymbol{x}_i$ is treated as a supplier holding $\frac{1}{n}$ units of goods while the j-th label needs q_{ij} units of goods. To obtain pseudo-labels, we take a re-scaled solution $n\boldsymbol{Q}$ bearing an assumption of a uniform marginal $\frac{1}{n}$ of $\boldsymbol{x}_i$.

With further scrutiny of the objective function above, this formulation complies with two constraints we posited, such that: (i)-it avoids biased assignment towards (only) the head labels in a long-tailed PLL setup, with $\boldsymbol{Q}^{\top}\mathbf{1}_n = \boldsymbol{r}$ constraining the distribution of pseudo-labeling to match the prior of the

class labels. (ii)-the probability mass of $\boldsymbol{Q}$ in each row is driven to be distributed within the candidate label set of a sample $\boldsymbol{x}_i$.

In Eq. (1), we use $-\log(p_{ij})$ as cost such that predictions with higher confidence enjoy a lower cost. This can also be replaced by other cost functions $h(p_{ij})$ as long as h is monotonically decreasing. But, we show that the negative log-likelihood cost results in a statistically consistent risk estimator.

Theorem 1. *Denote $\tilde{\Delta}$ the set of measures on $\mathcal{X} \times \mathcal{Y}$ that satisfies the constraints in Eq. (1). Let $\mathcal{D}_p$ be the distribution of tuple $(\boldsymbol{x}, S)$. The population risk of our method is as follows,*

$$\mathcal{R}(f) = \inf_{\varpi \in \tilde{\Delta}} \mathbb{E}_{(\boldsymbol{x},S)\sim\mathcal{D}_p} KL(f(\boldsymbol{x})||\varpi(y|\boldsymbol{x})). \tag{2}$$

Let f^{**} and f^{*} denote the optimal classifiers trained on fully-supervised data with cross-entropy loss and PLL data with $\mathcal{R}(f)$, respectively. Assume the small ambiguity condition [5] is satisfied. Then, under the deterministic scenario and ignoring the null set, we have $f^{*} = f^{**}$.

We refer the readers to Appendix A.1 for the proof. Theorem 1 indicates that the optimal classifier f^{**} can be recovered by our algorithm at a population level. Compared with other theoretically consistent PLL risks [6, 16], our risk preserves the marginal class distribution of the estimated pseudo-labels, and is reasonably more favorable in the long-tailed learning setup.

Fast Approximation. To resolve Eq. (1), we adapt the well-known Sinkhorn-Knopp algorithm [13] for efficient optimization. Formally, we define a matrix $\boldsymbol{M}$ such that $m_{ij} = p_{ij}^{\lambda}\mathbb{I}(j \in S_i)$, where $\lambda > 0$ is a smoothing regularization coefficient. The pseudo-labels are obtained by,

$$\begin{gathered} \boldsymbol{Q} = n \cdot \text{diag}(\boldsymbol{\alpha})\boldsymbol{M}\text{diag}(\boldsymbol{\beta}), \\ \text{with iteratively updated} \quad \boldsymbol{\alpha} \leftarrow \boldsymbol{c}./(\boldsymbol{M}\boldsymbol{\beta}), \quad \boldsymbol{\beta} \leftarrow \boldsymbol{r}./(\boldsymbol{M}^{\top}\boldsymbol{\alpha}). \end{gathered} \tag{3}$$

Here $./$ denotes element-wise division. $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\boldsymbol{\beta} \in \mathbb{R}^L$ are known as scaling coefficients vectors. Detailed derivation are provided in Appendix A.2. Here, we set $m_{ij} = 0$ if $j \notin S_i$, which is equivalent to define $\log(0) = -\inf$. Thus, $\boldsymbol{Q}$ will also be assigned 0s outside the candidate sets.

Inspired by [17], we also involve a queue acceleration trick to avoid traveling the whole training set. We found that Eq. (3) typically converges to a satisfactory result in < 50 rounds. With only simple algebra operations, this can be efficiently implemented on GPUs; see Appendix C.3.

3.2 Iterative Class Prior Estimation

Unlike conventional long-tailed learning scenarios [14, 15], the PLL setup is presented with label ambiguity in the proximity of candidate sets. This makes it intractable to estimate the marginal class prior distribution, especially by counting training samples grouped by class labels.

In SoLar, towards estimating the class marginals, our aim is to propose a solution with (almost) full autonomy that can adapt to the data distribution and the training state of the model. Here, we propose two interdependent techniques to achieve this goal. The resulted estimation is utilized in Eq (1). The core of our method is two-fold: (i)-we propose a moving-average style updating rule for class prior estimation, resulting in *stable training dynamics*; (ii)-we carefully devise a sample selection mechanism for robust training, while *preserving the marginal label distribution* as much as possible.

Moving-Average Distribution Updating. We propose using the model predicted results as a proxy for class prior estimation. In spite of its simplicity, we cannot ignore the fact during the early stage of training, the predicted labels can be largely imprecise, not trustworthy, and biased towards the head labels. Therefore, we propose an update mechanism in a moving-average fashion. Since we hold no assumption on the true class prior, the updating scheme is initiated from a uniform class prior $\boldsymbol{r} = [1/L, \ldots, 1/L]$. Respectively at each cycle (often defined by epochs), SoLar refines the distribution as follows.

$$\boldsymbol{r} \leftarrow \mu\boldsymbol{r} + (1-\mu)\boldsymbol{z}, \quad \text{where } z_j = \frac{1}{n}\sum\nolimits_{i=1}^{n} \mathbb{I}(j = \arg\max\nolimits_{j' \in S_i} f_{j'}(\boldsymbol{x}_i)), \tag{4}$$

where $\mu \in [0, 1]$ is a preset scalar. As displayed, the class prior is progressively updated alongside stabler training dynamics through a linear function. As the training proceeds, the model becomes more accurate and thus the estimated distribution becomes more reliable.

Class-wise Reliable Sample Selection. In the early stages of training, the class prior estimation can be inaccurate due to the poor pseudo label predictions. Thus, we posit that the lack of proper sample selection at early stages may jeopardize the marginal prior estimation. As common as it is for PLL learning dynamics, we want to reduce the variance of the pseudo-label quality, via a reliable sampling algorithm. This module works together with the moving-average class prior estimation.

A brute-force but effective strategy is to employ the small-loss sampling criterion in noisy-label learning [18], which assumes low-quality pseudo-labels can likely be characterized by a large loss. However, in a long-tailed setup, the tail labels can also incur a large loss because of their low-shot nature, making them rarely selected. We empirically show this phenomenon in Section 4.3. We hence implement the following procedures into the proposed SoLar framework. The goal of such a module is to make sure the coverage of the tail labels as well as to keep the marginal prior unbiased.

To begin with, to make sure that all labels — particularly the tail labels — are coverd, we split the batch of data B into L slices that are indexed by the *argmax* taken from the refined pseudo-labels: $B_j = \{(\boldsymbol{x}_i, \boldsymbol{q}_i) | j = \arg\max_{j' \in S_i} q_{ij'}\} \subset B$. In what follows, we derive two combinatorial rules on the selectivity of the training samples:

(i) Based on the aforementioned setup, we modify the small-loss criterion in the noisy-label learning [18] and adapt it to a long-tailed setup. Specifically, we first compute the instance-wise loss by marginalizing over the class index: $l_i = -\sum_{j=1}^{L} q_{ij} \log(p_{ij})$. Then, within the provided set B_j, we anchor the total sample number to be $k = \min(\lceil r_j \rho |B| \rceil, |B_j|)$, which is associated with r_j for *distribution-preserving*. The samples are being selected with top-k smallest l_i.

(ii) A parallel mechanism is inspired by the application of pseudo-labeling schemes in semi-supervised learning [19]. To adopt such approaches in SoLar, we purposely select those samples having a very high confidence score which is measured by the cosine similarity: $e_i = \boldsymbol{q}_i^\top \boldsymbol{p}_i$. As such, this second criteria selects samples that meet the inequality $e_i > \tau$. This mechanism further ensures high utility on the (likely) well-disambiguated samples.

Notice that, $\rho, \tau > 0$ are two thresholding hyper-parameters. The two introduced mechanisms are established to work collaboratively. Simply put, samples that satisfy either criterion are used for training the classifier. We will show later in Section 4.3 the efficacy of our sample selection mechanisms.

4 Experiments

In this section, we experimentally analyze the proposed SoLar method under various scenarios for the imbalanced PLL problem. More empirical results can be found in Appendix C. *The source code of SoLar is attached in the Supplementary.*

4.1 Setup

Datasets. First, we evaluate SoLar on two long-tailed datasets CIFAR10-LT and CIFAR100-LT introduced in [20, 21]. The training images are randomly removed class-wise to follow a pre-defined imbalance ratio $\gamma = \frac{n_1}{n_L}$, where n_j is the image number of the j-th class. For convenience, class indices are sorted based on the class-wise sample size, in descending order with $n_1 \geq ... \geq n_L$. We have $n_1 = 5,000$ for CIFAR10-LT and $n_1 = 500$ for CIFAR100-LT. We use different imbalance ratios to evaluate the performance of SoLar, with $\gamma \in \{50, 100, 200\}$ for CIFAR10-LT and $\gamma \in \{10, 20, 50\}$ for CIFAR100-LT. We then generate partially labeled datasets by manually flipping negative labels $\bar{y} \neq y$ to false-positive labels with probability $\phi = P(\bar{y} \in Y | \bar{y} \neq y)$, which follows the settings in previous works [6, 22]. The final candidate label set is composed of the ground-truth label and the flipped false-positive labels. We choose $\phi \in \{0.3, 0.5\}$ for CIFAR10-LT and $\phi \in \{0.05, 0.1\}$ for CIFAR100-LT. For all experiments, we report the mean and standard deviation based on 3 independent runs (with different random seeds).

Baselines. We compare SoLar with six state-of-the-art partial-label learning methods: 1) PiCO [7] leverages contrastive learning to disambiguate the candidate labels by updating the pseudo-labels with contrastive prototype labels. 2) PRODEN [6] is also a pseudo-labeling method that iteratively updates the latent label distribution by re-normalized classifier prediction. 3) VALEN [23] assumes the candidate labels are instance-dependent and recovers the latent label distributions by a Bayesian

Table 1: Accuracy comparisons on CIFAR10-LT and CIFAR100-LT under various flipping probability ϕ and imbalance ratio γ. Bold indicates superior results.

Methods	CIFAR10-LT					
	$\phi = 0.3$			$\phi = 0.5$		
	$\gamma = 50$	$\gamma = 100$	$\gamma = 200$	$\gamma = 50$	$\gamma = 100$	$\gamma = 200$
MSE	61.13±1.08	52.59±0.48	48.09±0.45	49.61±1.42	43.90±0.77	39.52±0.70
EXP	52.93±3.44	43.59±0.16	42.56±0.44	50.62±3.00	43.69±2.72	41.07±0.62
LWS	44.51±0.03	43.60±0.12	42.33±0.58	24.62±9.67	27.33±1.84	28.74±1.86
VALEN	58.34±1.05	50.20±6.55	46.98±1.24	40.02±1.88	37.10±0.88	36.61±0.57
CC	78.76±0.27	71.86±0.78	63.38±0.79	73.09±0.40	64.88±1.03	54.41±0.85
PRODEN	81.95±0.19	71.09±0.54	63.00±0.54	66.00±3.60	62.17±3.36	54.65±1.00
PiCO	75.42±0.49	67.73±0.64	61.12±0.67	72.33±0.08	63.25±0.64	53.92±1.64
SoLar (ours)	**83.80**±0.52	**76.64**±1.66	**67.47**±1.05	**81.38**±2.84	**74.16**±3.03	**62.12**±4.33

Methods	CIFAR100-LT					
	$\phi = 0.05$			$\phi = 0.1$		
	$\gamma = 10$	$\gamma = 20$	$\gamma = 50$	$\gamma = 10$	$\gamma = 20$	$\gamma = 50$
MSE	49.92±0.64	43.94±0.86	37.77±0.40	42.99±0.47	37.19±0.72	31.49±0.35
EXP	25.86±0.94	24.84±0.40	23.58±0.47	24.82±1.41	21.27±1.24	19.88±0.43
LWS	48.85±2.16	35.88±1.29	19.22±8.56	6.10±2.05	7.16±2.03	5.15±0.36
VALEN	49.12±0.58	42.05±1.52	35.62±0.43	33.39±0.65	30.67±0.11	24.93±0.87
CC	60.36±0.52	54.33±0.21	45.83±0.31	57.91±0.41	51.09±0.48	**41.74**±0.41
PRODEN	60.31±0.50	50.39±0.96	42.29±0.44	47.32±0.60	41.82±0.55	35.11±0.08
PiCO	54.05±0.37	46.93±0.65	38.74±0.11	46.49±0.46	39.80±0.34	34.97±0.09
SoLar (ours)	**64.75**±0.71	**56.47**±0.76	**46.18**±0.85	**61.82**±0.71	**53.03**±0.56	40.96±1.01

parametrization model. 4) LWS [22] also works in a pseudo-labeling style, which weights the risk function by considering the trade-off between losses on candidate labels and non-candidate ones. 5) CC [16] is a classifier-consistent method that assumes the candidate label set is uniformly sampled. 6) MSE and EXP [24] utilize mean square error and exponential loss as the risk estimators. All the hyper-parameters are searched according to the original papers.

Implementation details. We use an 18-layer ResNet as the feature backbone. The model is trained for 1000 epochs using a standard SGD optimizer with a momentum of 0.9. The initial learning rate is set as 0.01, and decays by the cosine learning rate schedule. The batch size is 256. These configurations are applied for SoLar and all baselines for fair comparisons. For SoLar, we devise a pre-estimation training stage, where we run a model for $100/20$ epochs (on CIFAR10/100-LT) respectively to obtain a coarse-grained class prior. After that, we re-initialize the model weights and run with this class prior. For our Sinkhorn-Knopp algorithm, we fix the smoothing regularization parameter as $\lambda = 3$ and the length of the queue for acceleration as 64 times batch size. The moving-average parameter μ for class prior estimation is set as $0.1/0.05$ in the first stage and fixed as 0.01 later. For class-wise reliable sample selection, we linearly ramp up ρ from 0.2 to $0.5/0.6$ in the first 50 epochs and fix the high-confidence selection threshold τ as 0.99. To improve the representation ability of SoLar, we further involve consistency regularization [19] and Mixup [25] techniques on reliable examples; see Appendix B.1 for more details. For fair comparisons, we equip all the baselines except PiCO with these two techniques as well.

4.2 Main Results

SoLar achieves SOTA results. As shown in Table 1, SoLar significantly outperforms all the rivals by a notable margin under various settings of imbalance ratio γ and label ambiguity degree ϕ. Specifically, on CIFAR10-LT dataset with $\phi = 0.5$, we improve upon the best baseline by **8.29**%, **9.28**%, and **7.47**% when the imbalance ratio γ is set to 50, 100, and 200 respectively. Moreover, most baselines display significant performance degradation as ϕ increases, whereas SoLar remains competitive. We also verify the effectiveness of SoLar on a more challenging CIFAR100-LT dataset with 10 times more classes and thus stronger label ambiguity. Notably, the performance gap between SoLar and baselines remains substantial. An interesting observation is that CC obtains impressive

Table 2: Different shots accuracy comparisons on CIFAR10-LT ($\phi = 0.5, \gamma = 100$) and CIFAR100-LT ($\phi = 0.1, \gamma = 20$). The best results are marked in bold and the second-best marked in underline.

Methods	CIFAR10-LT			CIFAR100-LT		
	Many	Medium	Few	Many	Medium	Few
MSE	81.11	42.03	9.18	57.46	37.57	16.53
EXP	92.64	39.75	0.00	60.35	3.99	0.00
LWS	89.09	1.52	0.00	20.80	0.86	0.00
VALEN	85.30	28.78	0.00	58.74	16.25	0.07
CC	94.56	65.61	<u>34.22</u>	73.03	<u>52.15</u>	<u>28.05</u>
PRODEN	**96.83**	<u>72.18</u>	14.17	**76.86**	43.14	5.43
PiCO	93.33	66.14	29.30	70.75	42.42	6.14
SoLar (ours)	<u>96.50</u>	**76.01**	**49.34**	<u>74.33</u>	**54.09**	**30.62**

Table 3: Ablation results on CIFAR10-LT ($\phi = 0.5, \gamma = 100$) and CIFAR100-LT ($\phi = 0.1, \gamma = 20$).

Ablation	CIFAR10-LT				CIFAR100-LT			
	All	Many	Med.	Few	All	Many	Med.	Few
SoLar	74.16	96.50	76.01	49.34	53.03	74.33	54.09	30.62
PRODEN w S+CPE	62.83	96.80	69.79	19.58	41.69	75.70	42.29	7.06
SoLar w/o Pre-Est	63.97	79.97	70.21	39.67	50.51	68.09	54.15	29.18
SoLar w Cand-Est	74.34	96.37	74.47	52.14	52.78	76.15	53.93	28.22
SoLar-Oracle	74.11	96.81	77.10	47.43	54.16	77.91	56.94	27.55
SoLar w/o S	29.61	86.10	6.95	3.34	35.80	66.85	32.06	8.61
SoLar w GS	53.01	96.58	60.10	0.00	33.62	72.03	23.91	5.21
SoLar w/o S-SL	47.49	96.46	46.37	0.00	12.80	30.42	7.97	0.15

performance on the CIFAR-100 dataset, but is still inferior to SoLar in most cases. These observations clearly validate the superiority of SoLar.

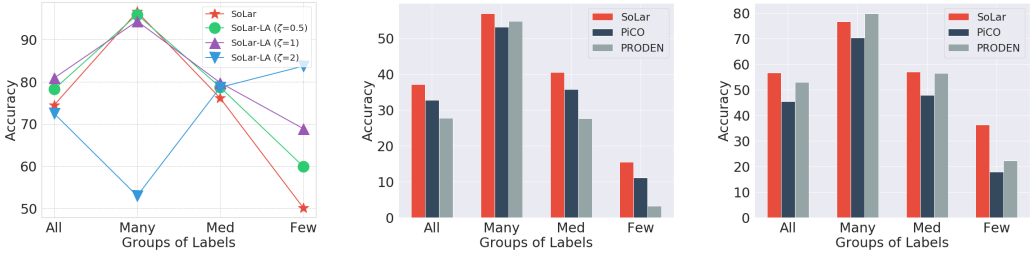
Results on different groups of labels. We show that SoLar achieves overall strong performance on *both* head and tail classes. To see this, we report accuracy on three groups of classes with different sample sizes. Recall from Section 4.1 that the class indices are sorted based on the sample size, in descending order. We divide the classes into three groups: many ($\{1, 2, 3\}$), medium ($\{4, 5, 6, 7\}$), and few ($\{8, 9, 10\}$) shots for CIFAR10-LT and many ($\{1, \ldots, 33\}$), medium ($\{34, \ldots, 67\}$), and few ($\{68, \ldots, 100\}$) shots for CIFAR100-LT. Table 2 shows the accuracy of different groups on both CIFAR10-LT and CIFAR100-LT. On few-shot classes, the gaps between SoLar and the best baseline are **15.12**% and **2.57**% on CIFAR10-LT and CIFAR100-LT. Moreover, SoLar retains a competitive performance on many-shot labels (*i.e.*, head classes). In contrast, most baselines exhibit degenerated performance on tail classes, especially on the CIFAR100-LT dataset. The comparisons highlight that SoLar poses a good disambiguation ability in the long-tailed PLL setup.

4.3 Ablation Studies

In this section, we present our main ablation analysis to show the effectiveness of SoLar. The main results of our ablation studies are shown in Table 3; more results can be found in Appendix C.4.

Effect of Sinkhorn label refinery. It is of interest to see whether the performance improvement comes from our Sinkhorn label refinery procedure. To see this, we equip PRODEN with our sample selection and class prior estimation mechanism (dubbed *PRODEN w S+CPE*). As shown in Table 3, PRODEN w S+CPE still fails to disambiguate labels on the tail, which validates the superiority of our Sinkhorn label refinery procedure.

Effect of class prior estimation. To verify the effectiveness of class prior estimation in SoLar, we compare with two variants: 1) *SoLar w/o Pre-Est* which removes the class prior pre-estimation stage of SoLar and trains from uniformed class prior; 2) *SoLar w Cand-Est* initializes the class prior distribution by counting per-class candidate numbers (*i.e.*, $r_j = \sum_{i=1}^{n} \mathbb{I}(j \in S_i)$); 3) *SoLar-Oracle* also removes the pre-estimation stage but trains from the real class prior. The results are in Table 3. We draw two salient observations. First, SoLar outperforms SoLar w/o Pre-Est, which suggests that



(a) SoLar with logit adjustment (b) CUB200-LT (c) CIFAR100-H-LT

Figure 2: (a) Performance comparisons of SoLar and SoLar with logit adjustment (SoLar-LA) on CIFAR10-LT ($\phi = 0.5, \gamma = 100$). ζ is the temperature parameter of logit adjustment. (b) Performance comparisons of SoLar and baselines on the fine-grained CUB200-LT dataset. (c) Performance comparisons of SoLar and baselines on the CIFAR100-LT dataset with hierarchical labels (CIFAR100-H-LT).

using coarsely-estimated class prior is more beneficial than using uniform class prior. Secondly, SoLar w Cand-Est . Finally, we contrast with SoLar-Oracle, which theoretically serves as an upper bound by using the real class prior. We observe that SoLar's performance favorably matches or even outperforms SoLar-Oracle on few-shot classes. The results are encouraging since SoLar does not require access to real class prior at all.

Effect of sample selection. Next, we experiment with different sample selection strategies, including: 1) *SoLar w/o S* which regards all examples as clean samples and does not perform selection; 2) *SoLar w GS* which performs selection in a global manner rather than class-wise; 3) *SoLar w/o S-SL* which only selects high-confidence examples. From Table 3, we can observe that SoLar w/o S worsened the performance because many pseudo-labels are unreliable at the beginning. SoLar w/o GS works relatively well on data-rich labels but suffers on the tail. This happens because the majority classes dominate the sample selection. SoLar w/o S-SL performs similarly to SoLar w/o GS on CIFAR10-LT, but diverges on CIFAR100-LT. In the beginning, few examples are selected as the classifier is less confident, resulting in degenerated solution. Our results indicate that our sample selection procedure is indispensable to SoLar in handling long-tailed class distribution.

4.4 Further Analysis

Combining SoLar with logit adjustment. While our work mainly focuses on label disambiguation from the PLL perspective, potential improvement is straightforward by applying long-tailed learning techniques to well-disambiguated datasets. To this end, we equip SoLar with *logit adjustment* [14] (dubbed *SoLar-LA*), a state-of-the-art LTL technique, for improved performance. Denote the logits of our model by $g(\cdot)$, *i.e.*, the output of the network prior to the softmax layer. Given a testing data $\boldsymbol{x}$, SoLar-LA made prediction by $\arg\max_{j \in [L]} g_j(\boldsymbol{x}) - \zeta \cdot \log(r_j)$, where ζ is a temperature parameter for calibration. This post-hoc corrected prediction is known to be theoretically consistent for minimizing the balanced error [14]. Note that logit adjustment cannot be applied to the original PLL data directly, as it requires the true class prior as well as a classifier trained by minimizing the long-tailed empirical risk. Fortunately, SoLar achieves both goals. Hence, we implement SoLar-LA by using the classifier as well as the estimated class prior of SoLar. Figure 2 (a) reports the performance of SoLar-LA with varying ζ values. With a proper ζ, SoLar-LA improves upon SoLar by **6.72**% on CIFAR10-LT. This suggests that SoLar is compatible with existing LTL methods and opens the door to exploring more advanced LTL techniques.

Results on fine-grained partial-label learning. In practice, semantically similar classes can lead to significant label ambiguity, as exemplified in Figure 1 (a). To test the limit of SoLar, we follow PiCO [7] and evaluate on two fine-grained datasets: 1) CUB200-LT [26] dataset with 200 bird species; 2) CIFAR100-LT with hierarchical labels (CIFAR100-H-LT), where the candidate labels are generated within the same superclass[1]. We set $\phi = 0.05, \gamma = 5$ for CUB200-LT and $\phi = 0.5, \gamma = 20$ for CIFAR100-H-LT. In Figure 2 (b) and (c), we compare SoLar with two strongest baselines PRODEN and PiCO, where SoLar improves the best baselines by a large margin (**4.37**% on CUB200-LT and **3.67**% on CIFAR100-H-LT). These results clearly validate the effectiveness of SoLar, even when the dataset presents severe label ambiguity.

[1]The CIFAR100 dataset comprises 20 superclasses, with 5 classes in each superclass.

Results on real-world long-tailed learning data. Additionally, we conduct experiments on the real-world long-tailed SUN397 dataset [27] containing 108,754 RGB images and 397 scene classes. We hold out 50 samples per class for testing and the resulting training set has an imbalanced ratio of $2311/50 \approx 46$. Similarly, we synthesize a partial-label dataset set with an ambiguity degrees of $\phi = 0.05$; see Appendix C.1 for more details. From Table 4, we can observe that SoLar still outperforms the baselines by a substantial margin.

Table 4: Performance comparisons on the SUN397 dataset with $\phi = 0.05$.

Methods	All	Many	Medium	Few
PRODEN	51.14	76.98	58.17	17.61
PiCO	29.54	57.91	21.27	9.33
SoLar	**61.58**	**77.36**	**62.59**	**44.49**

5 Related Work

Partial-label Learning (PLL) [4] allows each example to be equipped with a candidate label set with the ground-truth label being concealed. A plethora of PLL methods [5, 28, 29, 30] have been developed. For example, maximum margin-based methods [31, 32] maximize the margins between candidate labels and the remaining ones. Graph-based approaches [33, 8, 9, 10] typically leverage nearest neighbors to smooth the logical label vectors. Recently, pseudo-labeling methods [16, 6, 22, 34, 7, 35] become increasingly prevalent due to their promising performance, *e.g.*, PRODEN [6] re-normalizes classifier's outputs and PiCO [7] updates pseudo-targets by using contrastively learned prototypes. However, most of these works overlook the fact that class distribution can be imbalanced. Our empirical studies demonstrate that state-of-the-art PLL methods fail to disambiguate true labels on the minority classes, which motivates us to explore the optimal transport-based label disambiguation technique for PLL.

Long-tailed Learning (LTL) assumes that the training data follow Zipf's law that few labels pose a large number of instances, which is observed in a wide range of applications [11]. To cope with the LTL problem, sampling-based methods [36, 37] re-balance different classes at a data level, *e.g.*, up-sampling on minority classes. Re-weighting methods [20, 14, 38, 39, 15] typically adjust the weights of instances or classes that calibrates the classifier predictions to be balanced. Some works also explore transfer learning [40, 41] or ensemble learning [42, 43] in LTL. A recent trend in LTL [44, 45] is to decompose representation and classifier learning. Nevertheless, these methods assume a full-supervision of training data, which is unrealistic in many real-world applications. It has motivated researchers to study the LTL problem in weak-supervision scenarios, such as semi-supervised learning [46, 21] and noisy-label learning [47, 48]. In this work, we explore a different formulation of LTL where only candidate labels are provided.

Optimal Transport (OT) [12] is originally proposed to depict the distance between two measures. Recently, it has drawn huge attention from different fields in machine learning, including semi-supervised learning [49], domain adaptation [50, 51], object detection [52], and generative models [53]. The most related one to our work is OTLM [15], which also applies OT to the supervised LTL problem. By assuming a known test class distribution, OTLM performs OT-based post-hoc correction on model predictions to obtain balanced results. In contrast, our work targets the long-tailed PLL problem and holds no assumption on class priors.

6 Conclusion

In this work, we present a novel Sinkhorn label refinery framework (SoLar) for a challenging imbalanced partial-label learning problem. Key to our method, we derive an optimal transport objective that refines the pseudo-labels to match the true class prior. Additionally, we tackle the non-trivial class prior estimation problem by a moving-average technique, which is further guarded by a sample selection mechanism for controllable model training. Empirically, SoLar can run without knowledge of the real class distribution. Comprehensive experiments show that SoLar improves baseline algorithms by a significant margin. As we collect the real-world applications, the partial-labeled tasks often arise together with a long-tailed class distribution. That is, from the perspective of human annotators, a long-tailed label is much harder to tag and that may subsequently lead to a partial-labeled candidate set. In the future, we hope to explore this line of research in broader tasks. We hope our work will draw more attention from the community towards a broader view of tackling the imbalanced partial-label learning problem.

References

[1] Ching-Hui Chen, Vishal M. Patel, and Rama Chellappa. Learning from ambiguously labeled face images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(7):1653–1667, 2018.

[2] Forrest Briggs, Xiaoli Z. Fern, and Raviv Raich. Rank-loss support instance machines for MIML instance annotation. In *SIGKDD*, pages 534–542. ACM, 2012.

[3] Gabriel Panis and Andreas Lanitis. An overview of research activities in facial age estimation using the FG-NET aging database. In *ECCV*, volume 8926 of *Lecture Notes in Computer Science*, pages 737–750. Springer, 2014.

[4] Eyke Hüllermeier and Jürgen Beringer. Learning from ambiguously labeled examples. *Intell. Data Anal.*, 10(5):419–439, 2006.

[5] Timothée Cour, Benjamin Sapp, and Ben Taskar. Learning from partial labels. *J. Mach. Learn. Res.*, 12:1501–1536, 2011.

[6] Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, and Masashi Sugiyama. Progressive identification of true labels for partial-label learning. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 6500–6510. PMLR, 2020.

[7] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. Pico: Contrastive label disambiguation for partial label learning. *ICLR*, 2022.

[8] Deng-Bao Wang, Li Li, and Min-Ling Zhang. Adaptive graph guided disambiguation for partial label learning. In *SIGKDD*, pages 83–91. ACM, 2019.

[9] Ning Xu, Jiaqi Lv, and Xin Geng. Partial label learning via label enhancement. In *AAAI*, pages 5557–5564. AAAI Press, 2019.

[10] Gengyu Lyu, Songhe Feng, Tao Wang, Congyan Lang, and Yidong Li. GM-PLL: graph matching based partial label learning. *IEEE Trans. Knowl. Data Eng.*, 33(2):521–535, 2021.

[11] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *CoRR*, abs/2110.04596, 2021.

[12] Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Found. Trends Mach. Learn.*, 11(5-6):355–607, 2019.

[13] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, pages 2292–2300, 2013.

[14] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *ICLR*. OpenReview.net, 2021.

[15] Hanyu Peng, Mingming Sun, and Ping Li. Optimal transport for long-tailed recognition with learnable cost matrix. In *International Conference on Learning Representations*, 2022.

[16] Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. Provably consistent partial-label learning. In *NeurIPS*, 2020.

[17] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

[18] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, pages 8536–8546, 2018.

[19] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020.

[20] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Aréchiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *NeurIPS*, pages 1565–1576, 2019.

[21] Chen Wei, Kihyuk Sohn, Clayton Mellina, Alan L. Yuille, and Fan Yang. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10857–10866. Computer Vision Foundation / IEEE, 2021.

[22] Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. Leveraged weighted loss for partial label learning. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 11091–11100. PMLR, 2021.

[23] Ning Xu, Congyu Qiao, Xin Geng, and Min-Ling Zhang. Instance-dependent partial label learning. *Advances in Neural Information Processing Systems*, 34:27119–27130, 2021.

[24] Lei Feng, Takuo Kaneko, Bo Han, Gang Niu, Bo An, and Masashi Sugiyama. Learning with multiple complementary labels. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 3072–3081. PMLR, 2020.

[25] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*. OpenReview.net, 2018.

[26] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[27] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pages 3485–3492. IEEE Computer Society, 2010.

[28] Li-Ping Liu and Thomas G. Dietterich. A conditional multinomial mixture model for superset label learning. In *NeurIPS*, pages 557–565, 2012.

[29] Yan Yan and Yuhong Guo. Partial label learning with batch label correction. In *AAAI*, pages 6575–6582. AAAI Press, 2020.

[30] Changchun Li, Ximing Li, Jihong Ouyang, and Yiming Wang. Detecting the fake candidate instances: Ambiguous label learning with generative adversarial networks. In *CIKM*, pages 903–912. ACM, 2021.

[31] Nam Nguyen and Rich Caruana. Classification with partial labels. In *SIGKDD*, pages 551–559. ACM, 2008.

[32] Haobo Wang, Yuzhou Qiang, Chen Chen, Weiwei Liu, Tianlei Hu, Zhao Li, and Gang Chen. Online partial label learning. In *ECML PKDD*, volume 12458 of *Lecture Notes in Computer Science*, pages 455–470. Springer, 2020.

[33] Min-Ling Zhang, Bin-Bin Zhou, and Xu-Ying Liu. Partial label learning via feature-aware disambiguation. In *SIGKDD*, pages 1335–1344. ACM, 2016.

[34] Ning Xu, Congyu Qiao, Xin Geng, and Min-Ling Zhang. Instance-dependent partial label learning. In *NeurIPS*, pages 27119–27130, 2021.

[35] Fei Zhang, Lei Feng, Bo Han, Tongliang Liu, Gang Niu, Tao Qin, and Masashi Sugiyama. Exploiting class activation value for partial-label learning. In *International Conference on Learning Representations*, 2022.

[36] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.

[37] Jonathon Byrd and Zachary Chase Lipton. What is the effect of importance weighting in deep learning? In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 872–881. PMLR, 2019.

[38] Seulki Park, Jongin Lim, Younghan Jeon, and Jin Young Choi. Influence-balanced loss for imbalanced visual classification. In *ICCV*, pages 715–724. IEEE, 2021.

[39] Mingchen Li, Xuechen Zhang, Christos Thrampoulidis, Jiasi Chen, and Samet Oymak. Autobalance: Optimized loss functions for imbalanced data. In *NeurIPS*, pages 3163–3177, 2021.

[40] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, pages 2537–2546. Computer Vision Foundation / IEEE, 2019.

[41] Jaehyung Kim, Jongheon Jeong, and Jinwoo Shin. M2m: Imbalanced classification via major-to-minor translation. In *CVPR*, pages 13893–13902. Computer Vision Foundation / IEEE, 2020.

[42] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *CVPR*, pages 10988–10997. Computer Vision Foundation / IEEE, 2020.

[43] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X. Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *ICLR*. OpenReview.net, 2021.

[44] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*. OpenReview.net, 2020.

[45] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *NeurIPS*, 2020.

[46] Jaehyung Kim, Youngbum Hur, Sejun Park, Eunho Yang, Sung Ju Hwang, and Jinwoo Shin. Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. In *NeurIPS*, 2020.

[47] Kaidi Cao, Yining Chen, Junwei Lu, Nikos Aréchiga, Adrien Gaidon, and Tengyu Ma. Heteroskedastic and imbalanced deep learning with adaptive regularization. In *ICLR*. OpenReview.net, 2021.

[48] Tong Wei, Jiang-Xin Shi, Wei-Wei Tu, and Yu-Feng Li. Robust long-tailed learning under label noise. *CoRR*, abs/2108.11569, 2021.

[49] Kai Sheng Tai, Peter Bailis, and Gregory Valiant. Sinkhorn label allocation: Semi-supervised classification via annealed self-training. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 10065–10075. PMLR, 2021.

[50] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(9):1853–1865, 2017.

[51] Ievgen Redko, Nicolas Courty, Rémi Flamary, and Devis Tuia. Optimal transport for multi-source domain adaptation under target shift. In *AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pages 849–858. PMLR, 2019.

[52] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. OTA: optimal transport assignment for object detection. In *CVPR*, pages 303–312. Computer Vision Foundation / IEEE, 2021.

[53] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 2017.

[54] Xiyu Yu, Tongliang Liu, Mingming Gong, and Dacheng Tao. Learning with biased complementary labels. In *ECCV*, volume 11205 of *Lecture Notes in Computer Science*, pages 69–85. Springer, 2018.

[55] Philip A. Knight. The sinkhorn-knopp algorithm: Convergence and applications. *SIAM J. Matrix Anal. Appl.*, 30(1):261–275, 2008.

[56] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *NeurIPS*, 2020.

[57] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical data augmentation with no separate search. *CoRR*, abs/1909.13719, 2019.

[58] Min-Ling Zhang and Fei Yu. Solving the partial label learning problem: An instance-based approach. In *IJCAI*, pages 4048–4054. AAAI Press, 2015.

[59] Wei Wang and Min-Ling Zhang. Partial label learning with discrimination augmentation. In *SIGKDD*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [N/A]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix D.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 4.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.
   (b) Did you mention the license of the assets? [Yes] See Appendix C.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

A Theoretical Proofs

A.1 Proof of Theorem 1

First, we provide the following lemma to show the consistency of the standard cross-entropy loss.

Lemma 1. *If the cross-entropy loss is used as loss function, the optimal classifier f^{**} that minimizes the population risk $R_{CE}(f) = \mathbb{E}_{\boldsymbol{x},y}[-\log(f_y(x))]$ satisfies $f_y^{**}(\boldsymbol{x}) = p(y = i|\boldsymbol{x})$.*

Proof. We provide a proof sketch in the sequel, and a similar result has been shown in [54, 16]. The cross-entropy loss leads to the following optimization problem,

$$\min_f -\sum_{i=1}^{L} p(y = i|\boldsymbol{x}) \log(f_i(\boldsymbol{x})) \quad \text{s.t.} \quad \sum_{i=1}^{L} f_i(\boldsymbol{x}) = 1. \tag{5}$$

By introducing a multiplier ξ, the corresponding Lagrangian is as follows,

$$\mathcal{L}(f, \xi) = -\sum_{i=1}^{L} p(y = i|\boldsymbol{x}) \log(f_i(\boldsymbol{x})) + \xi(\sum_{i=1}^{L} f_i(\boldsymbol{x}) - 1). \tag{6}$$

Setting the derivative to 0 yields that,

$$f_i^{**}(\boldsymbol{x}) = \frac{1}{\xi} p(y = i|\boldsymbol{x}), \quad \sum_{i=1}^{L} f_i^{**}(\boldsymbol{x}) = \frac{1}{\xi} \sum_{i=1}^{L} p(y = i|\boldsymbol{x}) = 1. \tag{7}$$

We conclude that $\xi = 1$ and $f_y^{**}(\boldsymbol{x}) = p(y = i|\boldsymbol{x})$ holds. □

Next, we provide the definition of the (theoretical) ambiguity degree [5] to establish the learnability of the PLL problem.

Definition 1. *Denote the distribution of triplet $(\boldsymbol{x}, y, S)$ by $\hat{\mathcal{D}}_p$, We define the ambiguity degree as,*

$$\kappa = \sup_{(\boldsymbol{x},y,S)\sim\hat{\mathcal{D}}_p, p(\boldsymbol{x},y)>0, \bar{y}\neq y} P(\bar{y} \in S|\boldsymbol{x}, y). \tag{8}$$

We say a data distribution satisfies the *small ambiguity degree condition* if $\kappa < 1$. This is a natural requirement. If $\kappa = 1$, there exists at least one label pair that always co-occur, and thus, it is impossible to find the optimal hypothesis given partial-labeled data.

Now, we provide the main proof sketch for Theorem 1. Note that we always seek an optimal joint probability matrix before model training, which is mainly designed for empirical measures of the data samples. At a population level, we aim to search for an optimal probability measure that meets the marginal constraints and candidate constraints. Note that we aim to minimize $E(\boldsymbol{Q}, \boldsymbol{P}) = \mathrm{KL}(\boldsymbol{Q}||\boldsymbol{P}) + H(\boldsymbol{Q})$ where $\mathrm{KL}(\cdot)$ is the Kullback–Leibler divergence and $H(\cdot)$ is an entropy regularizer. This gives rise to our population risk,

$$\mathcal{R}(f) = \inf_{\varpi\in\tilde{\Delta}} \mathbb{E}_{(\boldsymbol{x},S)\sim\mathcal{D}_p} \mathrm{KL}(f(\boldsymbol{x})||\varpi(y|\boldsymbol{x})). \tag{9}$$

Here we omit the entropy term $H(\varpi)$ as it serves as a regularizer. As the training labels are categorical, we may assume $H(p(y|x)) = 0$, and thus, the infimum still holds. Otherwise, we can offset it by setting the smoothing parameter $\lambda = 1$ in the Sinkhorn-Knopp approximation. When the pseudo-labels are fixed, this objective is exactly the cross-entropy loss whose target measure is ϖ.

To show the consistency, we first prove that f^{**} is the minimizer of $\mathcal{R}(f)$. Substituting f^{**} into $\mathcal{R}(f)$ and combining with Lemma 1, we obtain that,

$$\mathcal{R}(f^{**}) = \min_{\varpi\in\tilde{\Delta}} E_{(\boldsymbol{x},S)\sim\mathcal{D}_p} \mathrm{KL}(p(y|\boldsymbol{x})||\varpi(y|\boldsymbol{x})) = 0. \tag{10}$$

It is obvious that $\varpi = p(y|\boldsymbol{x})$ leads to the minimal KL-divergence.

On the other hand, we show that f^{**} is the unique solution. We assume there exists another hypothesis f' that minimizes Eq. (9) and holds a different prediction $y' \neq y$ from f on at least one instance $(\boldsymbol{x}, y, S)$. By the deterministic assumption, we have $\mathcal{R}(f') = 0$, which holds only if y' is invariably included in the candidate set S, *i.e.*, $P_{\hat{\mathcal{D}}_p}(y' \in S|\boldsymbol{x}, y) = 1$. Clearly, it violates the small ambiguity degree condition and causes a contradiction. Ignoring all the null set where $p(\boldsymbol{x}, y) = 0$, we conclude that the minimizer f^* of our population risk equals the fully-supervised f^{**}.

The above discussion indicates that our method poses good performance guarantees like existing PLL methods [6, 16]. A similar loss $\mathcal{R}(f) = \mathbb{E}_{(\boldsymbol{x},S)\sim\mathcal{D}_p} \min_{y\in S} -\log(f_y(\boldsymbol{x}))$ is also discussed in [6], which is known as *minimal loss*. Its main difference from our objective is that we seek the best probability measure instead of an example-wise minimum. Our risk preserves the marginal constraint of the estimated pseudo-labels and is reasonably more favorable in the long-tailed learning setup.

A.2 Derivation of the Sinkhorn-Knopp Iteration

In this section, we briefly introduce the derivation of the Sinkhorn-Knopp algorithm to ensure the integrity of our work.

Recall that Eq. (1) is a standard linear programming (LP) problem, and can be solved in polynomial time. But, considering the high volume of data points as well as potential large class space, common LP solvers typically become time-consuming. To this end, we investigate a smoothed version of this optimization problem for fast approximation. Technically, we further add a negative entropy regularization term to obtain the following objective,

$$\min_{\boldsymbol{Q}\in\Delta} \langle \boldsymbol{Q}, -\log(\boldsymbol{P})\rangle - \frac{1}{\lambda} H(\boldsymbol{Q}). \tag{11}$$

This entropy regularizer is derived from an optimization perspective and is different from the one in our derivation of $E(\boldsymbol{Q}, \boldsymbol{P}) = \mathrm{KL}(\boldsymbol{Q}||\boldsymbol{P}) + H(\boldsymbol{Q})$. The resulting objective becomes smoothing and convex, and thus, can be efficiently solved.

It is worth pointing out that in the PLL problem, we have a special constraint that no probability mass is assigned outside the candidate label sets. To avoid introducing another set of annoying Lagrange multipliers, we transform this constraint to infinity costs on non-candidate labels. That is, we define a cost matrix $\boldsymbol{T}$ such that $T_{ij} = -\log(p_{ij})\mathbb{I}(j \in S_i) + \inf \mathbb{I}(j \notin S_i)$ and here we assume $0 \cdot \inf = 0$. Now, we obtain the Lagrangian of the new optimization problem,

$$\mathcal{L}(\boldsymbol{Q}, \boldsymbol{u}, \boldsymbol{v}) = \langle \boldsymbol{Q}, \boldsymbol{T}\rangle - \frac{1}{\lambda} H(\boldsymbol{Q}) + \boldsymbol{u}^\top(\boldsymbol{Q}\mathbf{1}_L - \boldsymbol{c}) + \boldsymbol{v}^\top(\boldsymbol{Q}^\top \mathbf{1}_n - \boldsymbol{r}), \tag{12}$$

where $\boldsymbol{u}, \boldsymbol{v}$ are Lagrange multipliers. As the original optimization is convex, the solution has to satisfy the Karush-Khun-Tucker (KKT) conditions. Therefore, we have the following equations,

$$\frac{\partial \mathcal{L}(\boldsymbol{Q}, \boldsymbol{u}, \boldsymbol{v})}{q_{ij}} = T_{ij} + \frac{1}{\lambda}(\log(q_{ij}) + 1) + u_i + v_j = 0. \tag{13}$$

Let $\boldsymbol{M}$ be a matrix such that $m_{ij} = e^{-\lambda T_{ij}} = p_{ij}^{\lambda}\mathbb{I}(j \in S_i)$. We can get that,

$$q_{ij} = e^{-\frac{1}{2}-\lambda u_i} m_{ij} e^{-\frac{1}{2}-\lambda v_j}. \tag{14}$$

Thereby, solving the primal problem in Eq. (11) equals to finding the multipliers $\boldsymbol{u}$ and $\boldsymbol{v}$. Again, this is equivalent to get another two vectors $\boldsymbol{\alpha} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^L$ such that $\alpha_i = e^{-\frac{1}{2}-\lambda u_i}$ and $\beta_j = e^{-\frac{1}{2}-\lambda v_j}$, which are also known as scaling coefficients vectors. Then, we can get the following equation,

$$\boldsymbol{Q} = \mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{M}\mathrm{diag}(\boldsymbol{\beta}). \tag{15}$$

Recall that we enforce $\boldsymbol{Q}$ to meet the following constraints,

$$\begin{aligned} \boldsymbol{Q}\mathbf{1}_L &= \mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{M}\boldsymbol{\beta} = \boldsymbol{c}, \\ \boldsymbol{Q}^\top \mathbf{1}_n &= \mathrm{diag}(\boldsymbol{\beta})\boldsymbol{M}^\top\boldsymbol{\alpha} = \boldsymbol{r}. \end{aligned} \tag{16}$$

This gives rise to an alternative coordinate descent algorithm for updating the scaling coefficients,

$$\boldsymbol{\alpha} \leftarrow \boldsymbol{c}./(\boldsymbol{M}\boldsymbol{\beta}), \quad \boldsymbol{\beta} \leftarrow \boldsymbol{r}./(\boldsymbol{M}^\top\boldsymbol{\alpha}). \tag{17}$$

It is also known as the Sinkhorn-Knopp fixed point iteration. We refer the readers to [55] for its convergence properties. Empirically, we find that setting $\lambda = 3$ and running 50 steps are enough to get a satisfactory solution. The final step is to take a re-scaled the minimizer $n\boldsymbol{Q}$ in Eq. 15, since $\boldsymbol{Q}$ serves as a joint probability matrix and the posterior is calculated by $\frac{p(\boldsymbol{x},y)}{p(\boldsymbol{x})} = np(\boldsymbol{x}, y)$. Without loss of generality, we slightly abuse the notation $\boldsymbol{Q}$ to denote the obtained pseudo-labels.

The core of our algorithm is to replace our zero constraints on non-candidates with infinity cost, which equals defining $\log(0) = -\inf$. These infinity values are then mapped back to 0s in the Sinkhorn-Knopp iteration, ensuring the feasibility of the calculation.

B Practical Implementation

In this section, we describe several details of the practical implementation of SoLar.

B.1 Details of Representation Enhancement

A recent work PiCO [7] has shown that existing PLL methods are typically trapped in a *disambiguation-representation dilemma*, where the low-quality representation and imperfect label disambiguation mutually deteriorate each other. It can be even worse in the imbalanced setup because of low-shot examples on the tail. While PiCO pioneers the contrastive learning technique in PLL for superior performance, we show this can be achieved by a much simpler design. In contrast to the complicated network architecture in PiCO [7], we involve consistency training regularizer along with Mixup augmentation to improve the representation quality.

Consistency Regularization. Recently, consistency regularization (CR) has been widely applied in weakly-supervised learning [19], which assumes that a classifier should produce similar class probability for a sample and its local augmented copies. Motivated by this, we also incorporated CR into PLL. Given an image $\boldsymbol{x}_i$, we adopt two data augmentation modules SimAugment [56] and RandAugment [57] to obtain a weakly augmented image $\boldsymbol{x}_i^w$ and strongly augmented counterpart $\boldsymbol{x}_i^s$. During training, the weak one is utilized to produce pseudo-labels by our Sinkhorn label refinery procedure as well as selecting the reliable subset. Then, we define the the CR loss $l_{\text{cr}} = -\sum_{j=1}^{L} q_{ij} \log(f_j(\boldsymbol{x}_i^s))$ that is the cross-entropy loss on $\boldsymbol{x}_i^s$ and $\boldsymbol{q}_i$.

Mixup. We further incorporate mixup training for improved performance. Given a pair of weakly-augmented examples $\boldsymbol{x}_i^w$ and $\boldsymbol{x}_j^w$ in the reliable set, we create a virtual training example by linearly interpolating both,

$$\begin{aligned} \boldsymbol{x}_i^m &= \sigma \boldsymbol{x}_i^w + (1-\sigma)\boldsymbol{x}_j^w, \\ \boldsymbol{q}_i^m &= \sigma \boldsymbol{q}_i^w + (1-\sigma)\boldsymbol{q}_j^w, \end{aligned} \tag{18}$$

where $\sigma \sim \text{Beta}(\varsigma, \varsigma)$ and we simply set $\varsigma = 4$ without further tuning. Similarly, we define the mixup loss l_{mix} as the cross-entropy loss on $\boldsymbol{x}_i^m$ and $\boldsymbol{q}_i^m$.

In our implementation, we add CR and Mixup on reliable examples only. For the remaining examples, since their pseudo-labels are unreliable, we train them with the classical re-normalized PLL loss l_{rn} [6]. The final loss is defined by,

$$l_{\text{cls}} = \eta(l_{\text{ce}} + l_{\text{cr}} + l_{\text{mix}}) + (1-\eta)l_{\text{rn}}, \tag{19}$$

where l_{ce} is the original classification loss of weakly-augmented instances. We linearly ramp up η from 0 to 0.9 in the first 50 epochs, which helps warm up the classifiers.

B.2 Relaxed Solution for the Sinkhorn-Knopp Algorithm

Recall that we start from a uniform class prior. Empirically, most of our experiments run very well. But, in some cases, it can result in an unsolvable optimal transport objective. For example, if there are only a few examples holding one specific label as a candidate ($\ll \frac{n}{L}$), we have no hope to constrain the sum of pseudo-labels on this label to be the average number of instances. Thus, the Sinkhorn-Knopp iteration diverges.

In our implementation, once divergence occurs, we return a relaxed solution to the optimal transport problem. Concretely, we modify the matrix $\boldsymbol{M}$ by adding a small number $\epsilon = 1e^{-5}$ on its zero

Table 5: Full results on the SUN397 dataset.

Methods	SoLar ($\phi = 0.05$)				SoLar ($\phi = 0.1$)			
	All	Many	Medium	Few	All	Many	Medium	Few
PRODEN	51.14	76.98	58.17	17.61	35.96	76.62	29.48	1.40
PiCO	29.54	57.91	21.27	9.33	12.22	24.17	9.23	3.18
SoLar	**61.58**	**77.36**	**62.59**	**44.49**	**55.64**	**76.78**	**57.87**	**31.86**

Table 6: Accuracy comparisons of SoLar and SoLar with logits adjustment (SoLar-LA). The best results are marked in bold and the second-best marked in underline.

Methods	CIFAR10-LT ($\phi = 0.5, \gamma = 100$)				CIFAR100-LT ($\phi = 0.1, \gamma = 20$)			
	All	Many	Medium	Few	All	Many	Medium	Few
SoLar	74.16	**96.50**	76.01	50.16	53.03	**74.33**	54.09	30.62
SoLar-LA ($\zeta = 0.5$)	<u>78.23</u>	<u>95.91</u>	<u>78.70</u>	59.94	54.17	<u>73.05</u>	55.75	33.66
SoLar-LA ($\zeta = 1$)	**80.88**	94.35	**79.76**	<u>68.90</u>	**54.82**	71.08	<u>56.90</u>	<u>36.40</u>
SoLar-LA ($\zeta = 2$)	72.48	52.98	78.68	**83.71**	<u>54.37</u>	64.45	**57.70**	**40.86**

entries. In other words, we regard all negative labels as potential candidates but assign them very large costs. Now, $\boldsymbol{M}$ is an element-wise positive matrix, and the Sinkhorn fixed iteration process guarantees to converge. We can re-run the Sinkhorn-Knopp iteration to obtain a relaxed solution. Finally, we set the non-candidate labels as zero again, as they should not be assigned any probability mass. By then, our algorithm can safely run with a uniform distribution. This procedure is easy to implement and increases only $1\times$ computation at most. Moreover, as the class prior is estimated better and better, we typically do not need this relaxed solution anymore.

In practice, it is not likely that we are fully unknowledgeable of the class distribution. Thus, we can also use a good initialization to avoid the aforesaid problem. Empirically, we find that initializing with the ratio of candidate label number is also a good choice for SoLar.

C Additional Experimental Results

In this section, we report the additional empirical results of our proposed SoLar framework. All experiments are conducted on a workstation with 8 NVIDIA A6000 GPUs. The licenses of our employed datasets are unknown (non-commercial).

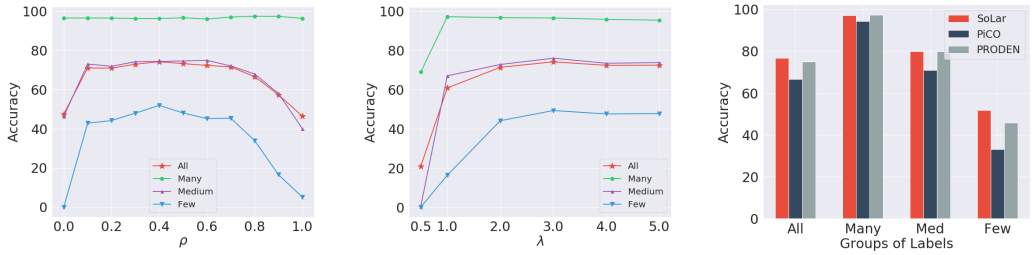
C.1 Experimental Setups on Fine-grained Datasets and SUN397

In the sequel, we show the full experimental setups on fine-grained classification datasets. In particular, on CUB200-LT, we set the batch size as 128, and the length of the queue for Sinkhorn acceleration as 8 times batch size. We train the model for 500 epochs without the pre-estimation training stage. The ratio parameter ρ ramps up from 0.2 to 0.5 in the first 50 epochs. Other hyper-parameters are the same as our default setting. On CIFAR100-H-LT, we simply adopt the default parameter configurations. The baselines are also fine-tuned to achieve their best results.

For the SUN397 dataset, we set the batch size as 128, and the length of the queue for Sinkhorn acceleration as 16 times batch size. We train the model for 20/200 epochs for distribution estimation and regular training. The gamma value is set as 0.1 at the pre-estimation stage. Other parameters are the same as the CUB200-LT dataset. As reported in Table 5, SoLar retains substantial performance advantages in different ambiguity degrees.

C.2 Full Results of SoLar with Logit Adjustment

In Table 6, we report the full results of SoLar-LA with varying ζ values on CIFAR10-LT and CIFAR100-LT. On CIFAR100-LT, SoLar-LA still outperforms SoLar with a proper ζ. The results demonstrate that SoLar achieves promising disambiguation ability in the imbalanced PLL setup.



(a) Ablation on ρ (b) Ablation on λ (c) Non-Uniform Case

Figure 3: (a) Performance of SoLar with varying ρ on CIFAR10-LT ($\phi = 0.5, \gamma = 100$). (b) Performance of SoLar with varying λ on CIFAR10-LT ($\phi = 0.5, \gamma = 100$). (c) Performance comparisons of SoLar and baselines on CIFAR10-LT with non-uniform generated candidate labels.

Given well-disambiguated data, SoLar makes it possible to apply off-the-shelf LTL methods for further improvements.

C.3 Running Time of the Sinkhorn-Knopp Algorithm

As we mentioned in Section 3.1, our Sinkhorn-Knopp algorithm can be efficiently implemented on GPUs. In Table 7, we report the total running time (1000 epochs) of the Sinkhorn-Knopp iterations as well as model training based on our implementation. We run SoLar with our default parameter configurations and evaluate using one NVIDIA A6000 GPU. During training, we maintain a queue of size 64×256 to store classifier predictions in previous 64 steps. Then, we concatenate the prediction in the current batch with the queue to run the Sinkhorn-Knopp algorithm. It can be shown that the Sinkhorn-Knopp iterations take less than $1/10$ time cost than regular model training (including forward pass and backward propagation). These results clearly validate the efficiency of our algorithm.

Table 7: Total running time (in hours) of the Sinkhorn-Knopp iterations and model training on CIFAR10-LT ($\gamma = 100$) and CIFAR100-LT ($\gamma = 20$).

Dataset	CIFAR10-LT	CIFAR100-LT
Sinkhorn-Knopp	0.08	0.10
Model Training	1.81	2.33

C.4 More Ablation

Effect of selection threshold ρ and τ. We further investigate the effect of the small-loss selection ratio parameter ρ and the high-confidence threshold τ. Figure 3 (a) shows the performance of SoLar with varying ρ on CIFAR10-LT (without ramp-up). When $\rho = 0$, SoLar simply selects high-confidence samples, which leads to an unsatisfactory performance on few-shot labels. The performance becomes much better as ρ becomes larger and achieves the best when $\rho = 0.4$. But, when ρ becomes too large, the model tends to overfit unreliable labels. Empirically, we find that SoLar works well in a wide range of ρ and $\rho \approx 0.5$ is a good choice.

Table 8 lists the result of SoLar with varying τ values. With a relatively small τ, SoLar collects too many unconfident examples, which typically hurts the robustness of model training. When $\tau = 0.99$, SoLar achieves rather promising results on different groups of labels. A surprising observation is SoLar without high-confidence selection, *i.e.*, $\tau = 1$, achieves slightly better results than SoLar on few-shot labels. We find the reason is that those well-disambiguated examples on the majority classes are overlooked, which poses an effect of down-sampling and benefits learning on the minority classes. As a negative effect, this sacrifices the performance of many-shot labels, which leads to an interesting trade-off. We empirically set $\tau = 0.99$ to retain relatively good performance on many-shot labels.

Table 8: Performance of SoLar with varying τ on CIFAR10-LT ($\phi = 0.5, \gamma = 100$).

τ Values	All	Many	Medium	Few
0.9	64.31	96.32	65.96	30.10
0.95	69.00	96.38	70.64	39.76
0.99	74.16	**96.50**	**76.01**	49.34
1.0	**74.49**	93.72	75.08	**54.46**

Effect of Sinkhorn smooth parameter λ. Next, we study the effect of the Sinkhorn smooth parameter λ. The results are shown in Figure 3 (b). In general, a too small λ results in poor label

Table 10: Ablation results on representation learning techniques.

Ablation	CIFAR10-LT				CIFAR100-LT			
	All	Many	Med.	Few	All	Many	Med.	Few
SoLar	74.16	96.50	76.01	49.34	53.03	74.33	54.09	30.62
CC w/o MU+CR	36.98	79.38	30.63	3.04	25.96	47.08	23.82	7.04
PRODEN w/o MU+CR	46.61	85.43	44.65	10.40	31.78	55.09	32.38	7.85
SoLar w/o MU	69.40	92.77	72.75	41.58	47.41	71.45	48.06	22.70
SoLar w/o CR	57.97	92.78	61.05	19.05	47.74	70.18	51.85	21.06
SoLar w/o MU+CR	44.83	82.33	40.39	13.25	30.88	50.52	30.53	11.61

assignment as the optimal transport objective becomes hard to be resolved. With a too large λ, our objective is over-smoothed and thus the resultant solution may deviate from the true one, which slightly degrades the performance as well. We empirically found that $\lambda = 3$ is a proper choice.

The role of unreliable examples. In our implementation, we enable unreliable examples to be trained with re-normalized PLL loss since their pseudo-labels can be noisy. This not only improves data utility but also serves as a warm-up mechanism for SoLar as we aim to train SoLar without knowledge of the true class prior. To see the role of unreliable examples, we also evaluate SoLar without l_{rn} and the results are shown in Table 9. We observe that SoLar w/o l_{rn} obtains favorable results on tail labels, even trained with only reliable examples. This verifies the importance of our distribution-preserving sample selection mechanism. Given the oracle class prior, SoLar w/o l_{rn} obtains competitive performance and matches our main results. In practice, proper utilization of unreliable samples typically leads to more promising results.

Table 9: Performance of SoLar without training with unreliable examples.

Class Prior	All	Many	Medium	Few
Oracle	71.75	96.00	76.65	40.97
Estimated	61.85	68.00	72.17	41.94

Ablation on representation learning. Here we ablate the contributions of two components in representation enhancement: mixup augmentation training and consistency regularization. Specifically, we compare SoLar with three variants: 1) *SoLar w/o MU* which removes Mixup augmentation training; 2) *SoLar w/o CR* which removes consistency regularization; 3) *SoLar w/o MU+CR* which removes both Mixup and consistency parts; 4) *PRODEN w/o MU+CR* is the PRODEN algorithm that removes Mixup and consistency parts as well. From Table 10, we can observe that both SoLar and PRODEN benefit from Mixup and consistency regularization techniques. In contrast to the relatively complicated contrastive learning modules in PiCO, we show that a simpler design can also alleviate the representation dilemma of PLL.

C.5 Results with Non-Uniform Data Generation

In reality, some labels may be more analogous to the ground-truth than others, and CIFAR100-H-LT is exactly one of the cases. Following [7, 22], we further test SoLar with a non-uniform data generation process, with the following flipping matrix:

$$\begin{bmatrix} 1 & 0.5 & 0.4 & 0.3 & 0.2 & 0.1 & 0 & \cdots & 0 \\ 0 & 1 & 0.5 & 0.4 & 0.3 & 0.2 & 0.1 & \cdots & 0 \\ \vdots & & & & \cdots & & & & \vdots \\ 0.5 & 0.4 & 0.3 & 0.2 & 0.1 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{20}$$

where each entry denotes the probability of a label being a candidate. From Figure 3 (c), we evaluate SoLar on CIFAR10-LT with an imbalance ratio $\gamma = 100$. It can be observed that SoLar still outperforms the baselines, which further validates its strong disambiguation ability.

C.6 Results on Real-world Partial-Label Learning Datasets

In this section, we test the performance of SoLar on four classical real-world datasets[2], including Lost, Bird Song, Soccer Player and Yahoo!News. As shown in Table 11, these datasets are naturally

[2]http://palm.seu.edu.cn/zhangml/Resources.htm#data

Table 11: Characteristics of the real-world partial label datasets. #Avg. Cand. indicates the average number of candidate labels per sample. Note that some labels in the Lost dataset are not related to any sample at all and hence, we calculate the imbalanced ratio by using the the non-zero min label count.

Datasets	#Examples	#Features	#Labels	#Avg. Cand.†	Imb. Ratio
BirdSong	4,998	38	13	2.18	11.33
Lost	1,122	108	16	2.23	40.00
Soccer Player	17,472	279	171	2.09	954.33
Yahoo! News	22,991	163	219	1.91	308.79

Table 12: Performance comparisons on real-world partial-label learning datasets.

Methods	Lost	BirdSong	Soccer Player	Yahoo!News
	Uniform Testing Set			
SoLar	77.86±6.36	**72.05**±1.76	**57.94**±1.13	67.62±0.64
VALEN	74.11±4.14	71.59±2.08	57.16±0.62	67.93±0.30
PRODEN	**77.98**±5.83	71.81±1.52	57.12±1.28	67.87±0.87
CC	77.85±5.83	71.86±1.94	56.44±0.90	**67.96**±0.90
IPAL	72.50±2.92	70.28±1.33	54.79±1.37	66.50±1.05
PLDA	66.07±1.89	67.68±1.94	50.26±0.46	53.66±0.99
	Balanced Testing Set			
SoLar	**70.56**±3.25	**68.72**±1.17	**24.97**±0.68	**58.18**±0.68
VALEN	60.95±2.71	67.49±1.22	20.56±0.79	56.30±0.70
PRODEN	68.85±4.17	67.72±1.42	24.22±0.88	55.98±0.67
CC	68.18±3.97	67.78±1.35	23.84±1.01	56.01±0.72
IPAL	64.35±2.50	66.82±1.49	11.34±0.23	54.90±0.53
PLDA	54.28±4.77	53.24±2.08	16.17±0.90	25.50±0.50

imbalanced. In particular, the Soccer Player dataset has an extremely severe imbalanced ratio of 954.33. We posit the evaluation protocols of previous works, *i.e.*uniformly split a testing set, are not realistic on these real-world datasets. To this end, we propose a balanced testing set sampling rule as follows: i) if one label is associated with > 200 data points, we uniformly select 100 samples for testing; ii) otherwise, we uniformly sample half of the data for testing. For performance comparisons, we additionally compare SoLar with two more PLL methods that are tailored for tabular data: IPAL [58] is a graph-based PLL method that propagates candidate labels to recover label confidences; PLDA [59] is a feature-selection-based method that maximizes the mutual-information-based dependency between features and labels and we choose PL-SVM [31] as the base learner. As shown in Table 12, SoLar achieves comparable or better performance to all the baselines under the conventional uniform testing set splitting setup. When the testing set is roughly balanced, SoLar obtains the best performance on all four datasets. These results further highlight the superiority of SoLar on the imbalanced PLL problem.

D Societal Impact

In this section, we briefly discuss several societal impacts of our work. The most obvious merit of our study is to reduce the cost of annotation by enabling coarse-grained labeling. This is a two-edged sword for the community. On the one hand, non-expert annotators can be employed for crowdsourcing labeling. On the other hand, if the partial-label learning paradigm is widely applied, the need for precisely annotated data would be significantly reduced, which may cause potential employment destruction as a consequence of reducing the need for human labelers. Another potential application of our method is data privacy. For instance, we may ask respondents to answer some private information when collecting some survey data. The candidate set-style labeling enables the respondents to exclude several wrong answers, which would be more privacy-friendly.