PICO: CONTRASTIVE LABEL DISAMBIGUATION FORPARTIAL LABEL LEARNING

Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, Junbo Zhao

Abstract

Partial label learning (PLL) is an important problem that allows each training example to be labeled with a coarse candidate set, which well suits many real-world data annotation scenarios with label ambiguity. Despite the promise, the performance of PLL often lags behind the supervised counterpart. In this work, we bridge the gap by addressing two key research challenges in PLL—representation learning and label disambiguation—in one coherent framework. Specifically, our proposed framework PiCO consists of a contrastive learning module along with a novel class prototype-based label disambiguation algorithm. PiCO produces closely aligned representations for examples from the same classes and facilitates label disambiguation. Theoretically, we show that these two components are mutually beneficial, and can be rigorously justified from an expectation maximization (EM) algorithm perspective. Extensive experiments demonstrate that PiCO significantly outperforms the current state-of-the-art approaches in PLL and even achieves comparable results to fully supervised learning.

Keywords：*Partial label learning, Contrastive learning*.

1 Introduction

The training of modern deep neural networks typically requires massive labeled data, which imposes formidable obstacles in data collection. Of a particular challenge, data annotation in the real-world can naturally be subject to inherent label ambiguity and noise. Labeling ambiguity is prevalent yet often overlooked in many applications, such as web mining[1] and automatic image annotation[2]. This gives rise to the importance of partial label learning (PLL)[3][4], where each training example is equipped with a set of candidate labels instead of the exact groundtruth label. This stands in contrast to its supervised counterpart where one label must be chosen as the ”gold”. Arguably, the PLL problem is deemed more common and practical in various situations due to its relatively lower cost to annotations.

Despite the promise, a core challenge in PLL is label disambiguation, i.e., identifying the groundtruth label from the candidate label set. Existing methods typically require a good feature representation[5][6][7], and operate under the assumption that data points closer in the feature space are more likely to share the same ground-truth label. However, the reliance on representations has led to a non-trivial dilemma—the inherent label uncertainty can undesirably manifest in the representation learning process—the quality of which may, in turn, prevent effective label disambiguation. To date, few efforts have been made to resolve this.

This paper bridges the gap by reconciling the intrinsic tension between the two highly dependent problems —representation learning and label disambiguation—in one coherent and synergistic framework. Our framework, Partial label learning with COntrastive label disambiguation (dubbedPiCO), produces closely aligned

representations for examples from the same classes and facilitates label disambiguation. Specifically, PiCO encapsulates two key components. First, we leverage contrastive learning (CL)[8] to partial label learning, which is unexplored in previous PLL literature. To mitigate the key challenge of constructing positive pairs, we employ the classifier's output and generate pseudo positive pairs for contrastive comparison (Section 3.1). Second, based on the learned embeddings, we propose a novel prototype-based label disambiguation strategy (Section 3.2). Key to our method, we gradually update the pseudo target for classification, based on the closest class prototype. By alternating the two steps above, PiCO converges to a solution with a highly distinguishable representation for accurate classification. Empirically, PiCO establishes state-of-the-art performance on three benchmark datasets, outperforming the baselines by a significant margin (Section 4) and obtains results that are competitive with fully supervised learning.

Theoretically, we demonstrate that our contrastive representation learning and prototype-based label disambiguation are mutually beneficial, and can be rigorously interpreted from an Expectation Maximization (EM) algorithm perspective (Section 5). First, the refined pseudo labeling improves contrastive learning by selecting pseudo positive examples accurately. This can be analogous to the E-step, where we utilize the classifier's output to assign each data example to one label-specific cluster. Second, better contrastive performance in turn improves the quality of representations and thus the effectiveness of label disambiguation. This can be reasoned from an M-step perspective, where the contrastive loss partially maximizes the likelihood by clustering similar data examples. Finally, the training data will be mapped to a mixture of von Mises-Fisher distributions on the unit hypersphere, which facilitates label disambiguation by using the component-specific label.

Our **main contributions** are summarized as follows:

1. (Methodology). To the best of our knowledge, our paper pioneers the exploration of contrastive learning for partial label learning and proposes a novel framework termed PiCO. As an integral part of our algorithm, we also introduce a new prototype-based label disambiguation mechanism, that leverages the contrastively learned embeddings.

2. (Experiments). Empirically, our proposed PiCO framework establishes the state-of-the-art performance on three PLL tasks. Moreover, we make the first attempt to conduct experiments on fine-grained classification datasets, where we show classification performance improvement by up to 9.61% compared with the best baseline on the$CUB-200$ dataset.

3. (Theory). We theoretically interpret our framework from the expectation-maximization perspective. Our derivation is also generalizable to other CL methods and shows the alignment property in CL (Wang & Isola, 2020) mathematically equals the M-step in center-based clustering algorithms.

2 Related works

2.1 Partial Label Learning(PLL)

Partial Label Learning (PLL) allows each training example to be annotated with a candidate label set, in which the ground-truth is guaranteed to be included. The most intuitive solution is averagebased methods[3][4][9],

which treat all candidates equally. However, the key and obvious drawback is that the predictions can be severely misled by false positive labels. To disambiguate the ground-truth from the candidates, identification-based methods[10], which regard the ground-truth as a latent variable, have recently attracted increasing attention; representative approaches include maximum margin-based methods[11], graph-based methods[9][12][13][7], and clustering-based approaches[5]. Recently, self-training methods[14][15][16] have achieved state-of-the-art results on various benchmark datasets, which disambiguate the candidate label sets by means of the model outputs themselves. But, few efforts have been made to learn high-quality representations to reciprocate label disambiguation.

2.2 Contrastive Learning (CL)

Contrastive Learning (CL)[17][18] is a framework that learns discriminative representations through the use of instance similarity/dissimilarity. A plethora of works has explored the effectiveness of CL in unsupervised representation learning[17][18]. Recently,[8] propose supervised contrastive learning (SCL), an approach that aggregates data from the same class as the positive set and obtains improved performance on various supervised learning tasks. The success of SCL has motivated a series of works to apply CL to a number of weakly supervised learning tasks, including noisy label learning[19][20], semi-supervised learning[21], etc. Despite promising empirical results, however, these works, lack theoretical understandings. Theoretically show that the CL favors alignment and uniformity, and thoroughly analyzed the properties of uniformity. But, to date, the terminology alignmentremains confusing; we show it inherently maps data points to a mixture of vMF distributions.

3 Method

In this section, we describe our novel Partial label learning with COntrastive label disambiguation (PiCO) framework in detail. In a nutshell, PiCO comprises two key components tackling the representation quality (Section 3.1) and label ambiguity respectively (Section 3.2). The two components systematically work as a whole and reciprocate each other.

The key challenge of PLL is to identify the ground-truth label from the candidate label set. During training, we assign each image x_i a normalized vector $s_i \in [0,1]^C$ as the pseudo target, whose entries denote the probability of labels being the ground-truth. The total probability mass of 1 is allocated among candidate labels in Y_i. Note that s_i will be updated during the training procedure. Ideally, s_i should put more probability mass on the (unknown) ground-truth label y_i over the course of training. We train a classifier $f : \mathcal{X} \to [0,1]^C$ using cross-entropy loss, with s_i being the target prediction. The per-sample loss is given by:

$$\mathcal{L}_{\text{cls}}\left(f; \boldsymbol{x}_i, Y_i\right) = \sum_{j=1}^{C} -s_{i,j} \log\left(f^j\left(\boldsymbol{x}_i\right)\right) \text{ s.t. } \quad \sum_{j \in Y_i} s_{i,j} = 1 \text{ and } s_{i,j} = 0, \forall j \notin Y_i, \tag{1}$$

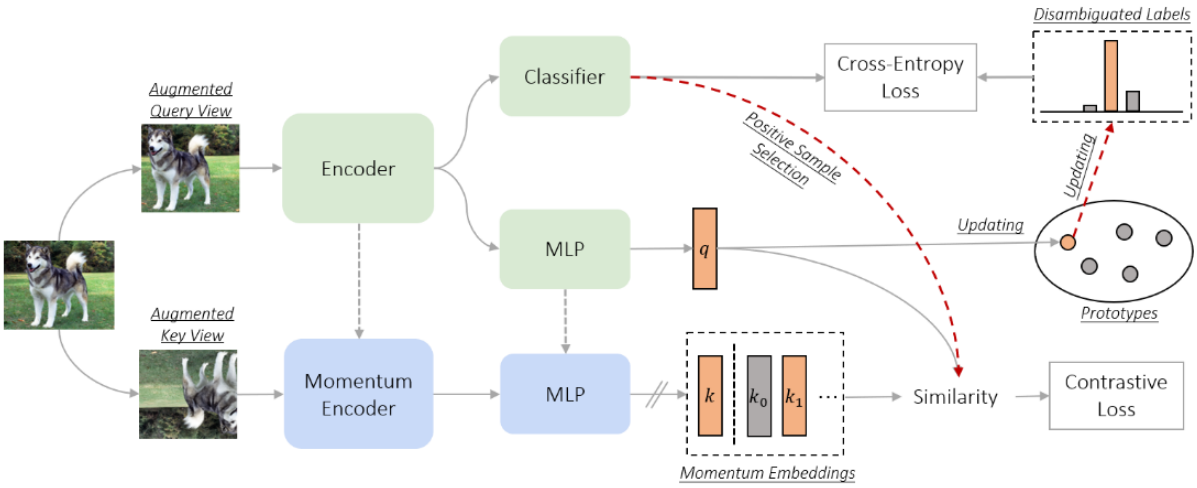


Figure 1: Illustration of PiCO. The classifier's output is used to determine the positive peers for contrastive learning. The contrastive prototypes are then used to gradually update the pseudo target. The momentum embeddings are maintained by a queue structure. '//' means stop gradient.

3.1 Contrastive Representation Learning for PLL

The uncertainty in the label space posits a unique obstacle for learning effective representations. In PiCO, we couple the classification loss in 1 with a contrastive term that facilitates a clustering effect in the embedding space. While contrastive learning has been extensively studied in recent literature, it remains untapped in the domain of PLL. The main challenge lies in the construction of a positive sample set. In conventional supervised CL frameworks, the positive sample pairs can be easily drawn according to the ground-truth labels[8]. However, this is not straightforward in the setting of PLL.

Training Objective. To begin with, we describe the standard contrastive loss term. We adopt the most popular setups by closely following MoCo and SupCon[8]. Given each sample (x, Y), we generate two views —a query view and a key view—by way of randomized data augmentation $Aug(x)$. The two images are then fed into a query network $g(\cdot)$ and a key network $g'(\cdot)$, yielding a pair of L2-normalized embeddings $\boldsymbol{q} = g\left(\text{Aug}_q(\boldsymbol{x})\right)$ and $k = g'\left(\text{Aug}_k(\boldsymbol{x})\right)$. In implementations, the query network shares the same convolutional blocks as the classifier, followed by a prediction head (see Figure 2). Following MoCo, the key network uses a momentum update with the query network. We additionally maintain a queue storing the most current key embeddings k, and we update the queue chronologically. To this end, we have the following contrastive embedding pool:

$$A = B_q \cup B_k \cup \text{ queue}, \tag{2}$$

where B_q and B_k are vectorial embeddings corresponding to the query and key views of the current mini-batch. Given an example x, the per-sample contrastive loss is defined by contrasting its query embedding with the remainder of the pool A,

$$\mathcal{L}_{\text{cont}}\left(g; \boldsymbol{x}, \tau, A\right) = -\frac{1}{|P(\boldsymbol{x})|} \sum_{\boldsymbol{k}_+ \in P(\boldsymbol{x})} \log \frac{\exp\left(\boldsymbol{q}^\top \boldsymbol{k}_+ / \tau\right)}{\sum_{\boldsymbol{k}' \in A(\boldsymbol{x})} \exp\left(\boldsymbol{q}^\top \boldsymbol{k}' / \tau\right)} \tag{3}$$

where $P(\boldsymbol{x})$ is the positive set and $A(\boldsymbol{x}) = A \backslash \{\boldsymbol{q}\}$, $\tau \geqslant 0$ is the temperature.

Positive Set Selection. As mentioned earlier, the crucial challenge is how to construct the positive set $P(x)$. We propose utilizing the predicted label $\tilde{y} = \arg\max_{j \in Y} f^j\left(\text{Aug}_q(\boldsymbol{x})\right)$ from the classifier. Note that we

restrict the predicted label to be in the candidate set Y . The positive examples are then selected as follows,

$$P(\boldsymbol{x}) = \{\boldsymbol{k}' \mid \boldsymbol{k}' \in A(\boldsymbol{x}), \tilde{y}' = \tilde{y}\} \tag{4}$$

where $\tilde{y}'$ is the predicted label for the corresponding training example of k'. For computational efficiency, we also maintain a label queue to store past predictions. In other words, we define the positive set of x to be those examples carrying the same approximated label prediction $\tilde{y}$. Despite its simplicity, we show that our selection strategy can be theoretically justified and also lead to superior empirical results. Putting it all together, we jointly train the classifier as well as the contrastive network. The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{cont}} \,. \tag{5}$$

Still, our goal of learning high-quality representation by CL relies on accurate classifier prediction for positive set selection, which remains unsolved in the presence of label ambiguity. To this end, we further propose a novel label disambiguation mechanism based on contrastive embeddings and show that these two components are mutually beneficial.

3.2 Prototype-based label disambiguation

As we mentioned, the contrastive loss poses a clustering effect in the embedding space. As a collaborative algorithm, we introduce our novel prototype-based label disambiguation strategy. Importantly, we keep a prototype embedding vector μc corresponding to each class c $\in \{1, 2, \ldots, \mathrm{C}\}$, which can be deemed as a set of representative embedding vectors. Categorically, a naive version of the pseudo target assignment is to find the nearest prototype of the current embedding vector. Notably this primitive resembles a clustering step. We additionally soften this hard label assignment version by using a moving-average style formula. To this end, we may posit intuitively that the employment of the prototype builds a connection with the clustering effect in the embedding space brought by the contrastive term (Section 3.1).

Pseudo Target Updating. We propose a softened and moving-average style strategy to update the pseudo targets. Specifically, we first initialize the pseudo targets with a uniform distribution, $s_j = \frac{1}{|Y|}\mathbb{I}(j \in Y)$. We then iteratively update it by the following moving-average mechanism,

$$\boldsymbol{s} = \phi \boldsymbol{s} + (1-\phi)\boldsymbol{z}, \quad z_c = \begin{cases} 1 & \text{if } c = \arg\max_{j \in Y} \boldsymbol{q}^\top \boldsymbol{\mu}_j \\ 0 & \text{else} \end{cases} \tag{6}$$

where $\varphi \in (0, 1)$ is a positive constant, and μj is a prototype corresponding to the j-th class. The intuition is that fitting uniform pseudo targets results in a good initialization for the classifier since the contrastive embeddings are less distinguishable at the beginning. The moving-average style strategy then smoothly updates the pseudo targets towards the correct ones, and meanwhile ensures stable dynamics of training. With more rigorous validation provided later in Section 5, we provide an explanation for the prototype as follows: (i)-for a given input x, the closest prototype is indicative of its ground-truth class label. At each step, s has the tendency to slightly move toward the one-hot distribution defined by z based on 6; (ii)-if an example consistently points to one prototype, the pseudo target s can converge (almost) to a one-hot vector with the least ambiguity.

Prototype Updating. The most canonical way to update the prototype embeddings is to compute it in

every iteration of training. However, this would extract a heavy computational toll and in turn cause unbearable training latency. As a result, we update the class-conditional prototype vector similarly in a moving-average style:

$$\boldsymbol{\mu}_c = \text{Normalize}\left(\gamma\boldsymbol{\mu}_c + (1-\gamma)\boldsymbol{q}\right), \quad \text{if } c = \arg\max_{j \in Y} f^{\mathcal{J}}\left(\text{Aug}_q(\boldsymbol{x})\right) \tag{7}$$

where the momentum prototype $\boldsymbol{\mu}_c$ of class c is defined by the moving-average of the normalized query embeddings q whose predicted class conforms to c. γ is a tunable hyperparameter.

3.3 Synergy between Contrast Learning and Label Disambiguation

While seemingly separated from each other, the two key components of PiCO work in a collaborative fashion. First, as the contrastive term favorably manifests a clustering effect in the embedding space, the label disambiguation module further leverages via setting more precise prototypes. Second, a set of well-polished label disambiguation results may, in turn, reciprocate the positive set construction which serves as a crucial part in the contrastive learning stage. The entire training process converges when the two components perform satisfactorily.

3.4 Method Improvement: Add Nosiy Partial Labels

In this section, we investigate a more practical setup called *noisy partial label learning*[22], where the true label potentially lies outside the candidate set, i.e., $y_i \notin Y_i$. Empirically, we observe that the current PLL methods, including PiCO, exhibit a significant performance drop on noisy PLL datasets. One main reason is that these methods mostly rely on a within-set pseudo-label updating step, but the noisy candidate sets in noisy PLL can mislead them towards overfitting on a wrong label.

So, we propose an extension of PiCO, which learns robust classifiers from noisy partial labels. First, we introduce a distance-based sample selection mechanism that detects *clean examples* whose candidate sets contain the ground-truth labels. Then, we develop a semi-supervised contrastive PLL framework to handle data with noisy candidates.

4 EXPERIMENTS

In this section, we present the main results and part of ablation results to show the effectiveness of PiCO and PiCO+ methods.

4.1 SETUP

Datasets. We evaluate PiCO on two commonly used benchmarks CIFAR-10 and CIFAR-100. And then generate partially labeled datasets by flipping negative labels $\bar{y} \neq y$ to false positive labels with a probability $q = P(\bar{y} \in Y \mid \bar{y} \neq y)$. In other words, all $C-1$ negative labels have a uniform probability to be false positive and we aggregate the flipped ones with the ground-truth to form the candidate set. We consider q $\in \{0.1, 0.3, 0.5\}$ for CIFAR-10 and q $\in \{0.01, 0.05, 0.1\}$for CIFAR-100. In Section 4.4, we further evaluate our method on fine-grained classification tasks, where label disambiguation can be more challenging. For the noisy PLL task, we introduce a noise controlling parameter $\eta = 1 - P(y \in Y|y)$ that controls the probability of the ground-truth label not being selected as a candidate. As it is possible that some instances are not assigned

any candidate and we simply re-generate the candidate set until it has at least one candidate label. We select $\eta \in \{0.1, 0.2\}$ for both datasets;

Implementation Details. we split a clean validation set (10% of training data) from the training set to select the hyperparameters. After that, we transform the validation set back to its original PLL form and incorporate it into the training set to accomplish the final model training. We use an 18-layer ResNet as the backbone for feature extraction. The projection head of the contrastive network is a 2-layer MLP that outputs 128-dimensional embeddings. We use two data augmentation modules SimAugmentmand RandAugment for query and key data augmentation respectively. Empirically, we find that even weak augmentation for key embeddings also leads to good results. The size of the queue that stores key embeddings is fixed to be 8192. The momentum coefficients are set as 0.999 for contrastive network updating and $\gamma = 0.99$ for prototype calculation. For pseudo target updating, we linearly ramp down ϕ from 0.95 to 0.8. The temperature parameter is set as $\tau = 0.07$. The loss weighting factor is set as $\lambda = 0.5$. The model is trained by a standard SGD optimizer with a momentum of 0.9 and the batch size is 256. We train the model for 800 epochs with cosine learning rate scheduling. We also empirically find that classifier warm up leads to better performance when there are many candidates. Hence, we disable contrastive learning in the first 100 epoch for CIFAR-100 with $q = 0.1$ and 1 epoch for the remaining experiments.

For PiCO+, we basically follow the original PiCO method. The clean sample selection ratio parameter δ is set as $0.8/0.6$ for noisy ratio $0.1/0.2$, respectively. For neighbor augmentation, we set $k = 16$ for CIFAR-10 and a smaller $k = 5$ for CIFAR-100. In the beginning, the embeddings can be less meaningful and thus, we enable kNN augmentation after the first 100 epochs. We fix the shape parameter of the Beta distribution to $\varsigma = 4$ for mixup training. We set the loss weighting factor $\alpha = 2$ and $\beta = 0.1$. Similar to the standard PLL setup, we warm up the model by fitting uniform targets for 5 and 50 epochs on CIFAR-10 and CIFAR-100 datasets respectively.

4.2 Experimental environment setup

The experiments are run on a Linux server with NVIDIA GeForce RTX 2080 Ti GPU with 11 GB Physical memory, and the Linux distribution Debian 10, x86 64 edition

4.3 Pseudo Code

Procedure 1 Pseudo-code of PiCO (one epoch).

Input: Training dataset $\mathcal{D}$, classifier f, query network g, key network g', momentum queue, uniform pseudo-labels $\boldsymbol{s}_i$ associated with $\boldsymbol{x}_i$ in $\mathcal{D}$, class prototypes $\boldsymbol{\mu}_j$ $(1 \leqslant j \leqslant C)$.

for $iter = 1, 2, \ldots,$ **do**
- sample a mini-batch B from $\mathcal{D}$
- `// query and key embeddings generation`
- $B_q = \{\boldsymbol{q}_i = g(\text{Aug}_q(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$
- $B_k = \{\boldsymbol{k}_i = g'(\text{Aug}_k(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$
- $A = B_q \cup B_k \cup \text{queue}$
- **for** $\boldsymbol{x}_i \in B$ **do**
  - `// classifier prediction`
  - $\tilde{y}_i = \arg\max_{j \in Y_i} f^j(\text{Aug}_q(\boldsymbol{x}_i))$
  - `// momentum prototype updating`
  - $\boldsymbol{\mu}_c = \text{Normalize}(\gamma \boldsymbol{\mu}_c + (1-\gamma)\boldsymbol{q}_i)$, if $\tilde{y}_i = c$
  - `// positive set generation`
  - $P(\boldsymbol{x}_i) = \{\boldsymbol{k}' | \boldsymbol{k}' \in A(\boldsymbol{x}_i), \tilde{y}' = \tilde{y}_i\}$
- **end**
- `// prototype-based label disambiguation`
- **for** $\boldsymbol{q}_i \in B_q$ **do**
  - $\boldsymbol{z}_i = \text{OneHot}(\arg\max_{j \in Y_i} \boldsymbol{q}_i^\top \boldsymbol{\mu}_j)$
  - $\boldsymbol{s}_i = \phi \boldsymbol{s}_i + (1-\phi)\boldsymbol{z}_i$
- **end**
- `// network updating`
- minimize loss $\mathcal{L}_{\text{pico}} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{cont}}$
- `// update the key network and momentum queue`
- momentum update g' by using g
- enqueue B_k and classifier predictions and dequeue

end

4.4 Main contributions

Our **main contributions** are summarized as follows:

1 (Practicality). Additionally, we propose PiCO+, an extension of PiCO, that targets to mitigate noisy candidate label sets. It further integrates a distance-based clean sample detection mechanism along with a semi-supervised contrastive learning module. We believe our work makes a serious attempt at improving the practicality of PLL in open-world environments.

2 (Experiments). Empirically, our proposed PiCO framework establishes the *state-of-the-art* performance on three PLL tasks. Moreover, we make the first attempt to conduct experiments on fine-grained classification datasets, where we show classification performance improvement by up to $\mathbf{9.61}\%$ compared with the best baseline on the CUB-200 dataset. We also evaluate our new PiCO+ framework on the noisy variants of these datasets, where PiCO+ outperforms the best baseline by up to $\mathbf{12.52}\%$ on the noisy PLL version of the CIFAR-10 dataset.

5 Results and analysis

Improvement: investigate a more practical setup called noisy partial label learning, where the true label potentially lies outside the candidate set, i.e., $y_i \in Y_i$. So we propose PiCO+, an extension of PiCO, which

learns robust classifiers from noisy partial labels. First, we introduce a distance-based sample selection mechanism that detects clean examples whose candidate sets contain the ground-truth labels. Then, we develop a semi-supervised contrastive PLL framework to handle data with noisy candidates.

5.1 Distance-based Clean Example Detection

Our motivation is that the clustering effect of contrastive learning makes the clean examples dominate the prototype calculation and thus they are distributed close to at least one prototype inside the candidate set. On the other hand, the noisy candidates mostly deviate from all candidate prototypes as their true labels are not contained in their candidate sets. So we propose a distance-based selection mechanism as follows,

$$\mathcal{D}_{\text{clean}} = \left\{(\boldsymbol{x}_i, Y_i) \mid \boldsymbol{q}_i^\top \boldsymbol{\mu}_{\tilde{y}_i} > \kappa_\delta\right\}$$

where $\tilde{y}_i = \arg\max_{j \in Y_i} f^j\left(\text{Aug}_a\left(\boldsymbol{x}_i\right)\right))$ is the classifier prediction. κ_δ is the $(100-\delta)$-percentile of the consine similarity between the query embedding and the $\tilde{y}_i$ prototype. For example, when δ = 60, it means 60% of examples are above the threshold.

5.2 Semi-supervised Contrastive Learning

We have made some improvements to the loss function. On clean datasets $\mathcal{D}_{clean}$, we assume the true labels are included in the candidate and run our PiCO method. On the noisy dataset, which is denoted by $\mathcal{D}_{noisy} = \mathcal{D}/\mathcal{D}_{clean}$, we follow our design patterns in PiCO to synergetically train the contrastive branch as well as the classifier. It is achieved by the following components:

Neighbor-Augmented Contrastive Learning. To this end, we first propose a label-driven construction approach. By regarding noisy samples as unlabeled data, it is intuitive to treat all labels as candidates. This gives rise to the following noisy positive set,

$$P_{\text{noisy}}\left(\boldsymbol{x}\right) = \{\boldsymbol{k}' \mid \boldsymbol{k}' \in A(\boldsymbol{x}), \hat{y}' = \hat{y}\}$$

$$\text{where } \hat{y} = \begin{cases} \arg\max_{1 \leqslant j \leqslant L} f^j\left(\text{Aug}_q(\boldsymbol{x})\right)\right) & \text{if } \boldsymbol{x} \in \mathcal{D}_{\text{noisy}} \\ \arg\max_{j \in Y} f^j\left(\text{Aug}_q(\boldsymbol{x})\right)\right) & \text{else.} \end{cases}$$

We apply this noisy positive set to all data inD and calculate a noisy contrastive loss Ln-cont by

$$\mathcal{L}_{\text{cont}}\left(g; \boldsymbol{x}, \tau, A\right) = -\frac{1}{|P(\boldsymbol{x})|} \sum_{\boldsymbol{k}_+ \in P(\boldsymbol{x})} \log \frac{\exp\left(\boldsymbol{q}^\top \boldsymbol{k}_+ / \tau\right)}{\sum_{\boldsymbol{k}' \in A(\boldsymbol{x})} \exp\left(\boldsymbol{q}^\top \boldsymbol{k}' / \tau\right)}$$

For the samples in the noise set, we also collect the nearest neighbors of noisy samples to be positive peers.

$$P_{\text{knn}}(\boldsymbol{x}) = \{\boldsymbol{k}' \mid \boldsymbol{k}' \in A(\boldsymbol{x}) \cap \mathcal{N}_k(\boldsymbol{x})\}$$

Then, we calculate the kNN-based contrastive loss $\mathcal{L}_{knn}$ on noisy examples. So the noisy examples are aligned to their local neighbors, which promotes their clustering effect towards the right labels.

Prototype-based Label Guessing. we leverage the class prototypes to guess their pseudo-targets s' by,

$$s'_j = \frac{\exp\left(\boldsymbol{q}^\top \boldsymbol{\mu}_j / \tau\right)}{\sum_{t=1}^{L} \exp\left(\boldsymbol{q}^\top \boldsymbol{\mu}_t / \tau\right)}, \quad \forall 1 \leqslant j \leqslant L$$

We calculate the cross-entropy loss on $\mathcal{D}_{noisy}$ as defined in

$$\mathcal{L}_{\text{cls}}\left(f;\boldsymbol{x}_i,Y_i\right)=\sum_{j=1}^{C}-s_{i,j}\log\left(f^j\left(\boldsymbol{x}_i\right)\right)\ \text{s.t.}\quad\sum_{j\in Y_i}s_{i,j}=1\ \text{and}\ s_{i,j}=0,\forall j\notin Y_i$$

, which is term as $\mathcal{L}_{n-cls}$.

Mixup Training. Recently, the mixup regularization technique has widely adopted to improve the robustness of weakly-supervised learning algorithms[23],[24]. Therefore, we also incorporate it into PiCO+ for boosted performance. Formally, given a pair of images xi and xj , we create a virtual training example by linearly interpolating both,

$$\boldsymbol{x}^m=\sigma\,\text{Aug}_q\left(\boldsymbol{x}_i\right)+(1-\sigma)\,\text{Aug}_q\left(\boldsymbol{x}_j\right)$$

$$\boldsymbol{s}^m=\sigma\hat{\boldsymbol{s}}_i+(1-\sigma)\hat{\boldsymbol{s}}_j$$

where $\sigma\sim\text{Beta}(\varsigma,\varsigma)$ and ς is a hyperparameter. Here, we take the pseudo-target of PiCO on clean examples, and the guessed label on noisy examples, i.e., $\hat{s}=s$ if $x\in\mathcal{D}_{\text{clean}}$ else $\hat{s}=s'$. We define the mixup loss $\mathcal{L}_{mix}$ as the cross-entropy on x^m and s^m .

Finally, we aggregate the above losses together,

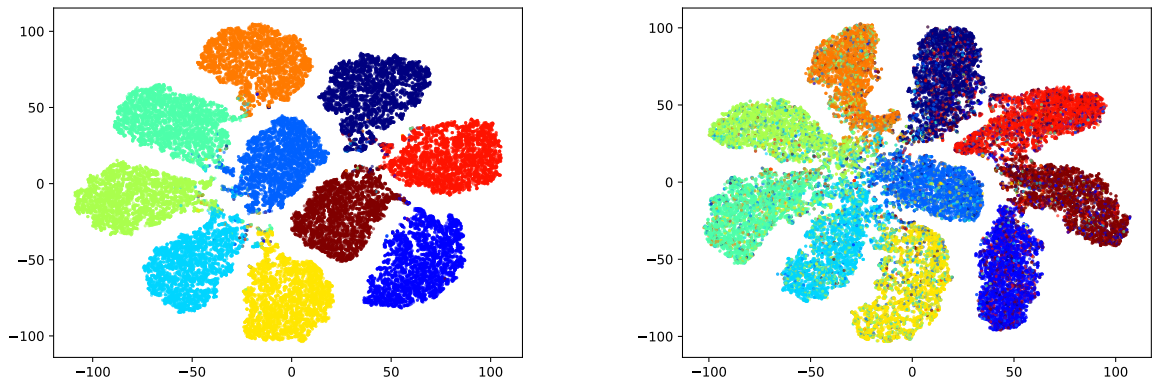
$$\mathcal{L}_{pico+}=\mathcal{L}_{mix}+\alpha\mathcal{L}_{clean}+\beta\left(\mathcal{L}_{n-cont}+\mathcal{L}_{knn}+\mathcal{L}_{n-cls}\right)$$

where $\mathcal{L}_{clean}$ is the PiCO loss on clean examples.

Experiment. The following is the final reproduction result:

Dataset	Method	$q=0.3$		$q=0.5$	
		$\eta=0.1$	$\eta=0.2$	$\eta=0.1$	$\eta=0.2$
CIFAR-10	PiCO+	**94.16 %**	**93.17 %**	**94.45 %**	**92.64 %**
	PiCO	89.81 %	84.54 %	88.09%	80.49%
Dataset	Method	$q=0.05$		$q=0.1$	
		$\eta=0.1$	$\eta=0.2$	$\eta=0.1$	$\eta=0.2$
CIFAR-100	PiCO+ (ours)	**74.68 %**	**72.40%**	**67.58 %**	**62.54 %**
	PiCO (ours)	66.50%	60.28%	53.77 %	47.80%

Table 1: Accuracy comparisons on noisy PLL datasets. Bold indicates superior results.



(a) PiCO features (b) PiCO+ features

Figure 2: T-SNE visualization of the image representation on CIFAR-10 ($q=0.5$). Different colors represent the corresponding classes.

PiCO+ learns compact and distinguishable features. In 2, we visualize the feature representations of PiCO+ on the noisy PLL CIFAR-10 dataset with $q = 0.5, \ = 0.2$. It can be shown that PiCO generates compact representations even with noisy candidate sets, which further supports the clustering effect of contrastive learning.

6 Conclusion and future work

In this work, we propose a novel partial label learning framework PiCO. The key idea is to identify the ground-truth from the candidate set by using contrastively learned embedding prototypes. Empirically, we conducted extensive experiments and show that PiCO establishes state-of-the-art performance. Our results are competitive with the fully supervised setting, where the ground-truth label is given explicitly. Theoretical analysis shows that PiCO can be interpreted from an EM-algorithm perspective. Additionally, we extend the PiCO framework to PiCO+ which is able to learn robust classifiers from noisy partial labels. Applications of multi-class classification with ambiguous labeling can benefit from our method, and we anticipate further research in PLL to extend this framework to tasks beyond image classification. We hope our work will draw more attention from the community toward a broader view of using contrastive prototypes for partial label learning.

References

[1] LUO J, ORABONA F. Learning from Candidate Labeling Sets[J]. neural information processing systems, 2010.

[2] CHEN C H, PATEL V M, CHELLAPPA R. Learning from Ambiguously Labeled Face Images[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017.

[3] HÜLLERMEIER E, BERINGER J. Learning from ambiguously labeled examples[J]. Intelligent Data Analysis, 2005.

[4] SAPP B, COUR T, TASKAR B. Learning from Partial Labels[J]. Journal of Machine Learning Research, 2011.

[5] LIU L P, DIETTERICH T G. A Conditional Multinomial Mixture Model for Superset Label Learning [J]. neural information processing systems, 2012.

[6] ZHOU B, ZHANG M L, LIU X Y. Partial Label Learning via Feature-Aware Disambiguation[J]. knowledge discovery and data mining, 2016.

[7] LYU G, FENG S, WANG T, et al. GM-PLL: Graph Matching based Partial Label Learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2019.

[8] KHOSLA P, TETERWAK P, WANG C, et al. Supervised Contrastive Learning[J]. neural information processing systems, 2020.

[9] ZHANG M L, YU F. Solving the partial label learning problem: an instance-based approach[J]. international conference on artificial intelligence, 2015.

[10] JIN R, GHAHRAMANI Z. Learning with Multiple Labels[J]. neural information processing systems, 2002.

[11] NGUYEN N, CARUANA R. Classification with partial labels[J]. knowledge discovery and data mining, 2008.

[12] WANG D B, LI L, ZHANG M L. Adaptive Graph Guided Disambiguation for Partial Label Learning [J]. knowledge discovery and data mining, 2019.

[13] XU N, LV J, GENG X. Partial Label Learning via Label Enhancement[J]. national conference on artificial intelligence, 2019.

[14] FENG L, LV J, HAN B, et al. Provably Consistent Partial-Label Learning[J]. neural information processing systems, 2020.

[15] LV J, XU M, FENG L, et al. Progressive Identification of True Labels for Partial-Label Learning[J]. international conference on machine learning, 2020.

[16] WEN H, CUI J, HANG H, et al. Leveraged Weighted Loss for Partial Label Learning[J]. international conference on machine learning, 2021.

[17] Van den OORD A, LI Y, VINYALS O. Representation Learning with Contrastive Predictive Coding[J]. arXiv: Learning, 2018.

[18] HE K, FAN H, WU Y, et al. Momentum Contrast for Unsupervised Visual Representation Learning[J]. arXiv: Computer Vision and Pattern Recognition, 2019.

[19] LI J, XIONG C, HOI S C H. MoPro: Webly Supervised Learning with Momentum Prototypes[J]. Learning, 2020.

[20] WU Z F, WEI T, JIANG J, et al. NGC: A Unified Framework for Learning With Open-World Noisy Data[J]. international conference on computer vision, 2021.

[21] YUHANG Z, ZHANG X, QIU R C, et al. Semi-supervised Contrastive Learning with Similarity Co-calibration.[J]. arXiv: Computer Vision and Pattern Recognition, 2021.

[22] LV J, FENG L, XU M, et al. On the Robustness of Average Losses for Partial-Label Learning[J]. CoRR, 2021, abs/2106.06152.

[23] BERTHELOT D, CARLINI N, GOODFELLOW I, et al. MixMatch: A Holistic Approach to Semi-Supervised Learning[J]. neural information processing systems, 2019.

[24] LI J, SOCHER R, HOI S C H. DivideMix: Learning with Noisy Labels as Semi-supervised Learning[J]. Learning, 2020.