MUTUAL PARTIAL LABEL LEARNING WITH COMPETITIVE LABEL NOISE

Anonymous authors
Paper under double-blind review

ABSTRACT

Partial label learning (PLL) is an important weakly supervised learning problem, where each training instance is associated with a set of candidate labels that include both the true label and noise labels. Most existing PLL methods assume the candidate noise labels are randomly chosen, which hardly holds in the real-world learning scenarios. In this paper, we consider a more realistic PLL scenario with competitive noise labels that are more difficult to distinguish from the true label than the random noise labels. We propose a novel Mutual Learning based PLL approach named ML-PLL to address this challenging problem. ML-PLL learns a prediction network based classifier and a class-prototype based classifier cooperatively through interactive mutual learning and label correction. Moreover, we use a transformation network to model the association relationships between the true label and candidate noise labels, and learn it together with the prediction network to match the observed candidate labels in the training data and enhance label correction. Extensive experiments are conducted on several benchmark PLL datasets, and the proposed ML-PLL approach demonstrates the state-of-the-art performance for partial label learning.

1 INTRODUCTION

As it is costly and difficult to annotate each instance with a precise label, weakly supervised learning (WSL) has been widely studied in recent years (Zhou, 2018), which includes, but not limited to, semi-supervised learning (Van Engelen & Hoos, 2020; Ouali et al., 2020), noise label learning (Natarajan et al., 2013; Feng et al., 2021), positive-unlabeled learning (Kiryo et al., 2017; Shu et al., 2020), and partial multi-label learning (Xie & Huang, 2018; Yan & Guo, 2021). Partial label learning (PLL) is a typical WSL problem and aims to learn a model from training samples with overcomplete labels; that is, each training sample is associated with a set of candidate labels that include both the true label and noise labels. PLL has been widely applied in many real-world learning scenarios, including automatic face naming (Hüllermeier & Beringer, 2006; Zeng et al., 2013), web mining (Luo & Orabona, 2010), and multimedia content analysis (Zeng et al., 2013).

Since the ground-truth label is hidden in the candidate label set and not available to the learning algorithms, the main challenge of PLL lies in candidate label disambiguation. To address this challenge, two main label disambiguation strategies have been proposed: average-based disambiguation strategy and identification-based disambiguation strategy. Average-based disambiguation treats each candidate label equally in the model training phase and averages all the modeling outputs from each candidate label in the testing phase(Cour et al., 2011; Hüllermeier & Beringer, 2006; Zhang & Yu, 2015). Although this strategy is simple and clear, it can make the true label overwhelmed by noise labels without sufficient differentiation, and lead to poor prediction performance. The identification-based disambiguation strategy treats the ground-truth label as a latent variable and tries to identify the true label by deriving different confidences for the candidate labels (Feng & An, 2018; 2019; Yao et al., 2020b; Yu & Zhang, 2016; Xu et al., 2021). The identification-based disambiguation approaches achieve relatively better prediction performance than the average-based disambiguation approaches by handling the candidate labels with discrimination, but they can still suffer from the potential drawback that the identified label may not be the true label, which could severely disrupt the subsequent model training. In addition, these existing methods are usually restricted to standard machine learning frameworks with liner or kernel-based approaches, which have difficulties in dealing with large-scale datasets.

Recently, deep learning has been widely used in dealing with PLL problems (Feng et al., 2020b; Wu et al., 2022). For example, the work in (Feng et al., 2020b) proposes two PLL methods, risk-consistent and classifier-consistent algorithms, based on deep neural networks. The methods in (Wen et al., 2021; Lv et al., 2020) try to progressively identify the ground-truth label during training procedures by employing a self-training technique. A more recent work in (Wang et al., 2022) tackles PLL by using contrastive representation learning with class-prototype based label disambiguation. Although it achieves satisfactory prediction performance, the contrastive learning procedure is usually time-consuming and resource-demanding. Moreover, all these methods have a common drawback: they automatically assume random noise in the label space; that is, they assume the candidate noise labels are randomly sampled from a uniform generating procedure. However, during the noise label creation procedure in real-world scenarios, such as crowd-sourcing, the noise labels are typically associated with the true label and dependent on the input sample, which makes the noise labels to be more difficult to distinguish from the ground-truth label than a random label.

In this paper, we consider a more realistic and challenging PLL learning scenario, where the noise labels are competitive and difficult to distinguish from the true label given the input data sample. Intuitively, competitive noise labels can demonstrate stronger association relationships with a true label than a random label noise, and hence are more likely to be chosen as candidate labels. For example, for online image annotation, when the object contained in an image is an "alpaca", a competitive label "camel" could have a large probability to be chosen by an annotator with limited expertise due to the similar appearances of "alpaca" and "camel", while labels such as "dog" or "duck" are less likely to be picked as part of the candidate label set due to their relatively weak association with the ground-truth "alpaca".

Motivated by the above consideration, we propose a novel and effective Mutual Partial Label Learning approach (ML-PLL) under the competitive label noise learning scenario, which learns a prediction network based classifier and class-prototype based classifier interactively through label correction and mutual learning. Specially, ML-PLL performs noise label correction by integrating the outputs of both the prediction network classifier and the class-prototype based classifier, while using the corrected pseudo-labels as the targets for the prediction network training and using the outputs of the prediction network as the targets for the prototype-based classifier training. In addition, a transformation network is proposed to model the association relationships between the true label and the noise candidate labels, which can further enhance the classifier training with respect to ground-truth label disambiguation. Extensive experiments are conducted on benchmarks PLL datasets, while the proposed approach ML-PLL demonstrates the state-of-the-art performance for partial label learning.

2 Related Work

2.1 Standard Partial Label Learning

The main challenge for dealing with PLL problem lies in how to disambiguate the candidate labels. Average-based disambiguation strategy and identification-based disambiguation strategy are two main strategies deployed in PLL. The average-based disambiguation strategy treats each candidate label equally in the model induction and averages all the modeling outputs from all the candidate labels as the final prediction. For example, the works in (Cour et al., 2011; Zhang & Yu, 2015) distinguish all the averaged candidate label prediction from the non-candidate ones. The identification-based disambiguation strategy treats the ground-truth label as a latent variable and tries to identify the true label by deriving confidence scores for all the candidate labels (Jin & Ghahramani, 2002; Feng & An, 2018; 2019; Yu & Zhang, 2016). For example, the works in (Zhang et al., 2016; Xu et al., 2019; Wang et al., 2021) try to identify the true label by employing iterative label refining procedures and leveraging the topological information in the feature space. However, these methods may suffer from the cumulative errors induced in the error-prone label confidence estimation along the topological structure. In addition, a number of PLL methods propose to employ the off-the-shelf learning techniques to tackle PLL problems. For the maximum likelihood technique, the likelihood of each PL training sample is defined over its candidate label set instead of its implicit ground-truth label (Jin & Ghahramani, 2002; Liu & Dietterich, 2012). For the k-nearest neighbor technique, the candidate labels from neighbor instances are aggregated to produce the final prediction on a test instance (Hüllermeier & Beringer, 2006; Gong et al., 2017; Zhang & Yu, 2015). For the maximum margin technique, the classification margin is defined over the predictive difference between

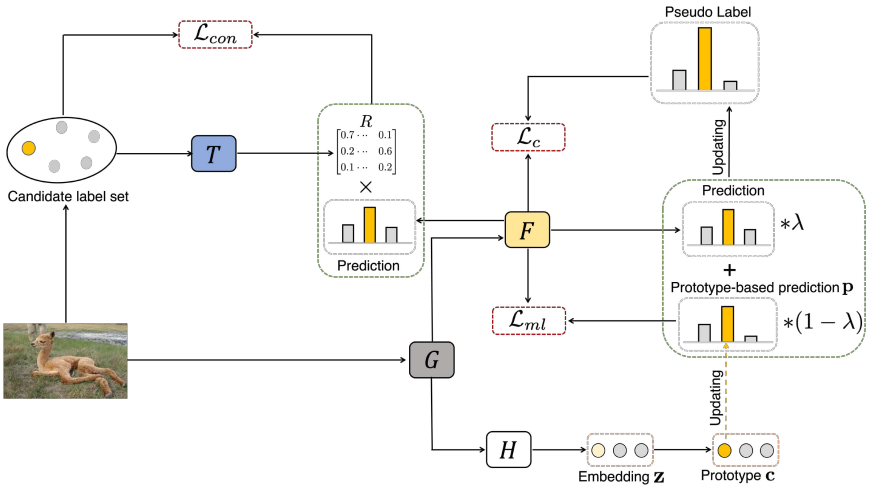


Figure 1: The proposed ML-PLL model, which learns the prediction network based classifier F and the class-prototype based classifier in a mutual learning manner. Moreover, the transformation network T is learned with the prediction network cooperately to match the observed candidate labels.

the candidate labels and the non-candidate labels for each PL training sample (Nguyen & Caruana, 2008)(Yu & Zhang, 2016). For the boosting technique, the weight of each PL training instance and the confidence value of each candidate label being the ground-truth label are refined in each boosting round (Tang & Zhang, 2017). For the error-correcting output codes (ECOC) technique, multiple binary classifiers corresponding to ECOC coding matrix are built based on the transformed binary training sets (Zhang et al., 2017).

2.2 Deep Neural Network based Partial Label Learning

Standard PLL methods are usually restricted to liner based models, which has difficulty in dealing with large-scale datasets. Deep learning paradigm based PLL has recently started gaining attention from the research community due to its powerful capability of training on large-scale datasets. The work in (Yao et al., 2020a) adopts deep convolutional neural networks as backbones and exploits the temporal-ensembling technique for model training. The work in (Yao et al., 2020b) proposes to train two networks in a mutual learning manner, which helps both networks interact with each other and alleviate the error accumulation problem. The work in (Lv et al., 2020) attempts to progressively identify the true label in a self-training manner, which can be compatibly trained with stochastic optimization. The work in (Feng et al., 2020b) assumes that the probability of each noise label being chosen is uniform and presents two provably consistent methods: a risk-consistent method and a classifier-consistent method. Another work in (Wen et al., 2021) induces a family of loss functions, which generalize the uniform generation procedures. In addition, a recent work in (Wang et al., 2022) handles PLL problems by conducting class prototype-based label disambiguation following the idea of contrastive representation learning (He et al., 2020; Li et al., 2020), which achieves impressive prediction performance.

Although the above approaches produce competitive performance, most of them assume the noise labels in the candidate label set are randomly produced, which however does not hold in many real-world learning scenarios. By contrast, we tackle a more realistic and challenging PLL scenario, where the noise candidate labels are competitive and difficult to distinguish from the true label.

3 Proposed Approach

In this section, we delineate the details of our proposed mutual partial label learning (ML-PLL) model in the competitive label noise scenario. Given a partial label training set $D = (X, Y) = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathcal{X}$ denotes the i-th instance and $\mathcal{X}$ denotes the input space; $\mathbf{y}_i \subseteq \mathcal{Y}$ denotes the candidate label indicator vector associated with $\mathbf{x}_i$, and $\mathcal{Y} = \{1, 2, \cdots, L\}$ denotes the

multi-class label space. The multiple 1 values in $\mathbf{y}_i$ denotes either the ground-truth label or irrelevant label noise, which forms the candidate label set S_i associated with $\mathbf{x}_i$. The task of PLL is to induce a multi-class classifier from the PLL training set D. We assume a challenging realistic PLL scenario, where the noise labels in the candidate label set are difficult to distinguish from the ground-truth label given the input instance.

The proposed ML-PLL model is illustrated in Figure 1, which mainly comprises the following component networks: a feature extractor G that extract features from the input instance $\mathbf{v} = G(\mathbf{x}) \in \mathbb{R}^m$; a projection network H that maps the feature vector $\mathbf{v}$ into a low-dimensional embedding $\mathbf{z} = H(\mathbf{v}) \in \mathbb{R}^d$; a prediction network based classifier F; a class-prototype based classifier; and a transformation network T that estimates the association relationships between the true label and candidate labels. The prediction network based classifier and the class-prototype based classifier are learned together through noise label correction and interactive mutual learning, while the prediction network and the transformation network are learned together to match the observed candidate labels in the training data and enhance label disambiguation. We elaborate these components and the proposed approach in the following subsections.

3.1 Noise Label Correction

The main challenge of PLL lies in that the ground-truth label coexists with the irreverent noise labels in the candidate label set and is unknown to the learning algorithm. The key of PLL is to correct the candidate label vectors and identify the true label vectors. Given an interactive pair of prediction network classifier F and prototype-based classifier, we propose the following simple mutual label correction procedure. For any training instance $\mathbf{x}_i$, we correct the partial label indicator vector towards the true label indicator vector by exploiting the predicted pseudo-label vector. Specifically, we produce the soft pseudo-label vector $q_i \in [0,1]^L$ for $\mathbf{x}_i$ by taking a weighted convex combination of the prediction network's softmax output probability vector $F(\mathbf{v}_i)$ and the class prototype-based prediction probability vector $\mathbf{p}_i$:

$$q_i = \lambda F(\mathbf{v}_i) + (1-\lambda)\mathbf{p}_i \tag{1}$$

where $\lambda \in (0,1)$ is the combination weight parameter. Here the prediction network $F(\mathbf{v}_i)$ is built above the feature extractor G, such as $F(\mathbf{v}_i) = F(G(\mathbf{x}_i))$, while $\mathbf{p}_i$ is produced by the similarity-based class-prototype classifier as follows:

$$p_{ik} = \frac{\exp(\mathbf{z}_i \cdot \mathbf{c}_k/\tau)}{\sum_{k=1}^{L} \exp(\mathbf{z}_i \cdot \mathbf{c}_k/\tau)} \tag{2}$$

where τ is a temperature parameter, L is the number of class categories, $\mathbf{z}_i = H(G(\mathbf{x}_i))$ denotes the low-dimensional embedding of the features for instance $\mathbf{x}_i$, and $\mathbf{c}_k$ denotes the prototype for the k-th category, which will be elaborated in the next subsection. Instead of using Euclidean distance, we use the dot-product operator "$\cdot$" to compute the inner product similarity between the instance and the class prototypes in the embedding space for class prediction probability calculation.

To maintain a effective soft pseudo-label vector for each instance and eliminate the noise predictions outside of the candidate label set, we restrict the soft pseudo-labels within the candidate label set and further re-normalize q_i into $\widehat{q}_i$ with row normalization,

$$\widehat{q}_{ik} = \begin{cases} \dfrac{q_{ik}}{\sum_{\mathbf{y}_{ij}>0} q_{ij}} & \text{if} \quad \mathbf{y}_{ik} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

A corrected soft pseudo-label matrix $\widehat{Q}$ can be obtained by performing the abovementioned noise label correction on all the n training instances, such that $\widehat{Q} = [\widehat{q}_{ik}]_{n\times L}$. With the progress of prediction network training and class-prototype update (elaborated in the next subsection), we expect the produced soft pseudo-label vector will be better aligned with the true label indicator vector, which can interactively provide a good target for the subsequent prediction network training and gradually mitigate the negative impact of the label noise during the training process.

3.2 Batch-wise Moving Average based Prototype Update

The effectiveness of noise label correction relies on accurate class probability predictions, which depend on both the prediction network and the class-prototypes. Instead of training an entirely

independent prototype-based classifier, we propose to produce the class prototypes based on the outputs of the prediction network classifier. Specifically, we compute the class prototypes in the low-dimensional embedding space from the partial label training set D in the following way:

$$\mathbf{c}_k = \frac{\sum_{\mathbf{x}_i \in D} H(G(\mathbf{x}_i)) * \mathbb{1}(\widehat{\mathbf{y}}_{ik} = 1)}{\sum_{i=1}^{n} \mathbb{1}(\widehat{\mathbf{y}}_{ik} = 1)}, \quad \text{for } \widehat{\mathbf{y}}_{ik} = \begin{cases} 1 & \text{if } k = \arg\max_{k \in S_i} F(G(\mathbf{x}_i)) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $\mathbb{1}$ is an indicator function, $\widehat{\mathbf{y}}_i$ denotes predicted one-hot label vector for $\mathbf{x}_i$, which is determined by the output of the prediction network F and restricted to the candidate label set S_i. In this way, we can automatically eliminate the prediction noise outside of the candidate labels.

With the prototype computing method above, we can update the class-prototype by recomputing it during each training iteration. However, such an update process will be computational-intensive and lead to intolerable training latency, since it requires going through the whole training set. To address this problem, we propose to perform batch-wise moving average based prototype update; that is, we update class-prototype for each mini-batch by consulting the cluster centroid over each mini-batch in a moving averaging manner:

$$\mathbf{c}_k = \text{Normalize}(\gamma \mathbf{c}_k + (1 - \gamma)\mathbf{c}'_k) \tag{5}$$

where $\mathbf{c}'_k$ is the mean embedding for the k-th class within the current mini-batch, which can be computed with Eq.(4) by limiting D to the current mini-batch; γ is a momentum coefficient; and $\text{Normalize}(\mathbf{c}) = \mathbf{c}/\|\mathbf{c}\|_2$. We expect such a batch-wise prototype update method can be computational efficient and enhance the gradual adjustment and stability of the class prototypes.

3.3 Prediction Network Training with Mutual Learning

With the noise label correction method above, the corrected soft pseudo-label produced can be treated as the target for prediction network training. Thus, we can learn the prediction network based probabilistic classifier on training data D by minimizing the following classification loss:

$$\mathcal{L}_c(F, G, H) = \frac{1}{n} \sum_{\mathbf{x}_i \sim D} \sum_{k=1}^{L} -\widehat{q}_{ik} \log(F^k(G(\mathbf{x}_i))) \tag{6}$$

where k is the index of the class, $\widehat{q}_i$ is the corrected soft pseudo-label vector for the i-th instance, and $F^k(\cdot)$ denotes the k-th entry of prediction network's softmax outputs, indicating the predicted probability of the instance belonging to the k-th class. It is expected that the prediction network based classifier F can be improved by training with the corrected soft pseudo-labels.

Meanwhile, we can learn an additional class-prototype based classifier supervised by the outputs of the prediction network by deploying the following interactive mutual learning loss:

$$\mathcal{L}_{ml}(F, G, H) = \frac{1}{n} \sum_{\mathbf{x}_i \sim D} \sum_{k=1}^{L} -F^k(G(\mathbf{x}_i)) \log(p_{ik}) \tag{7}$$

where p_{ik} is the prediction probability produced by prototype based classifier using Eq.(2). We expect this mutual learning loss can enforce both classifiers to collaborate with each other and gradually lead to consistent predictions that can better align with underlying true label indicator vectors.

3.4 Transformation based Label Association Learning

As previously discussed, the appearances of noise labels in the candidate label set typically have an association relationship with the ground-truth label and can be competitive to the true label given the input instance in the real-world learning scenarios. In this subsection, we propose to learn the competitive association relationship between noise candidate labels and the true label using a transformation network T, aiming to enhance the label disambiguation process. We assume that the transformation network T can be learned to produce a transformation matrix $R = [r_{ij}]_{L \times L}$, such that R can capture the transformation relationship from a given ground-truth label to the associated candidate noise labels and produce the candidate label set. The entry r_{ij} of R reflects the probability of transforming j-th label to a candidate label when the i-th label is the ground-truth. When r_{ij} is

large, the j-th label has a strong association relationship with the i-th ground-truth label and hence has a larger probability to be chosen as a candidate label.

To achieve the goal, we propose to produce the transformation matrix R by applying the transformation network on the observed candidate partial label matrix $Y = [\mathbf{y}_{ij}]_{n \times L}$ from the training set, such that $R = T(Y)$, since the observed partial label matrix contains both the true label and candidate noise labels. We propose to learn the transformation network T through prediction based candidate label matrix reconstruction. Specifically, we propose to reconstruct the normalized candidate label matrix by transforming the predicted outputs of the prediction network classifier F with the transformation matrix R. We use the symmetric Kullback-Leibler (KL) divergence loss to match the observed reconstructed candidate label distribution and the observed candidate label distribution through the following consistency loss:

$$\mathcal{L}_{con}(F, G, H, T) = \sum_{\mathbf{x}_i \sim D} \Big(\mathrm{KL}\big(F(G(\mathbf{x}_i))R, \widetilde{Y}_i\big) + \mathrm{KL}\big(\widetilde{Y}_i, F(G(\mathbf{x}_i))R\big) \Big) \tag{8}$$

where $\widetilde{Y}_i = Y_i / \|Y_i\|_2$ is a row normalized candidate label matrix in the training set.

It is however computational-intensive to recompute the transformation matrix R on the whole matrix Y. We propose to compute and update R on each mini-batch in a moving averaging style, such as

$$R = \eta R + (1 - \eta) R' \tag{9}$$

where R' is the transformation matrix computed from the observed partial label matrix within the current mini-batch, and η is a momentum smooth coefficient.

We expect the consistency loss in Eq.(8) can enhance the noise label correction and prediction network learning by exploiting the label association relationships between the true-label the candidate labels, thus consequently pushing the outputs of prediction network towards the true labels.

3.5 Learning with ML-PLL Model

By integrating the classification loss in Eq.(6), the mutual learning loss in Eq.(7) and the consistency loss in Eq.(8) together, we get the following overall training loss for proposed ML-PLL model,

$$\mathcal{L} = \mathcal{L}_c + \alpha \mathcal{L}_{ml} + \beta \mathcal{L}_{con} \tag{10}$$

where α and β are trade-off hyperparameters that control the relative importance of the mutual learning loss and the consistency loss respectively. For optimization, we minimize this objective using a mini-batch-based stochastic gradient descent (SGD) algorithm. The full training algorithm of ML-PLL is outlined in Appendix A.1.

4 Experiment

4.1 Experimental Setting

Datasets We perform the proposed ML-PLL on four widely used benchmark image datasets including Fashion-MNIST (Xiao et al., 2017), Kuzushiji-MNIST (Clanuwat et al., 2018), CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009). We manually corrupt these datasets into partially labeled version by competitive and uniform label noise (Lv et al., 2020) generation process. To synthesize the competitive candidate noise labels, we predict the probability of each category being the true label corresponding to an instance by using a clean neural network, which can be trained on the original clean datasets. Assume we have a clean training sample $(\mathbf{x}_i, \widehat{\mathbf{y}}_i)$, ρ_i is its prediction probability of instance $\mathbf{x}_i$, $\mathbf{y}_{ij}$ is the true label associated with $\mathbf{x}_i$. Firstly, we set the probability of j-th label to zero, i.e., ρ_{ij} = 0. Then, we sort ρ_i in a descending order and keep its original category index. Intuitively, the irrelevant labels with higher prediction values demonstrate the stronger association relationships with the true label, which can be chosen as the competitive labels and hence are more likely to be chosen as the candidate labels. In our experiments, we choose the irrelevant labels within the top-k predictions as the competitive labels, from which we randomly choose several competitive labels as the candidate labels that together with the true label $\widehat{\mathbf{y}}_{ij}$ to form the candidate label set associated with the instance $\mathbf{x}_i$. Following above competitive candidate noise label generation process, we corrupt the benchmark datasets Fashion-MNIST, Kuzushiji-MNIST and CIFAR-10

Table 1: Accuracy (mean±std) comparisons on Fashion-MNIST, Kuzushiji-MNIST and CIFAR-10 with competitive noise labels on different ambiguity levels. The best result is highlighted in bold.

Data set	Method	Avg.#CL = 3	Avg.#CL = 4	Avg.#CL = 5
Fashion-MNIST	ML-PLL	**90.54±0.06%**	**89.94±0.08%**	**89.37±0.09%**
	PiCO	90.06±0.07%	89.75±0.10%	88.98±0.00%
	LWS	89.31±0.05%	88.52±0.11%	81.07±0.05%
	CC	88.69±0.13%	88.23±0.10%	88.05±0.36%
	MSE	79.91±0.29%	74.01±0.25%	67.11±0.53%
	EXP	89.40±0.15%	89.15±0.10%	88.34±0.20%
Kuzushiji-MNIST	ML-PLL	**95.70±0.02%**	**95.38±0.07%**	**94.87±0.07%**
	PiCO	94.86±0.05%	94.13±0.07%	93.95±0.05%
	LWS	93.04±0.06%	92.40±0.05%	91.21±0.06%
	CC	94.58±0.32%	94.29±0.05%	93.10±0.06%
	MSE	64.99±0.25%	63.98±0.27%	61.29±0.22%
	EXP	93.73±0.15%	93.54±0.17%	92.04±0.20%
CIFAR-10	ML-PLL	**84.95±0.05%**	**83.52±0.06%**	**79.25±0.09%**
	PiCO	83.96±0.06%	82.04±0.08%	76.42±0.11%
	LWS	81.68±0.13%	79.23±0.16%	47.34±0.28%
	CC	73.12±0.44%	69.57±0.48%	62.11±0.43%
	MSE	53.43±0.19%	44.21±0.21%	33.52±0.33%
	EXP	73.93±0.23%	71.24±0.22%	51.96±0.28%

Table 2: Accuracy (mean±std) comparisons on CIFAR-100 with competitive noise labels on different ambiguity levels. The best result among each column is highlighted in bold.

Data set	Method	Avg.#CL = 2	Avg.#CL = 6	Avg.#CL = 10
CIFAR-100	ML-PLL	**61.39±0.10%**	**57.56±0.06%**	**53.53±0.08%**
	PiCO	59.86±0.10%	54.07±0.08%	49.22±0.05%
	LWS	55.00±0.20%	48.78±0.37%	36.99±0.40%
	CC	57.29±0.26%	51.00±0.29%	45.00±0.24%
	MSE	39.01±0.15%	29.76±0.11%	18.52±0.25%
	EXP	45.86±0.21%	40.38±0.27%	30.44±0.29%

from the top-6 predictions while CIFAR-100 from the top-20 predictions. The average number of candidate labels (Avg.#CL) and the characteristics of each benchmark dataset corrupted by the competitive label noise generation process is recorded in Appendix A.4. To synthesize the uniform candidate noise labels, we corrupt these datasets by the uniform generation procedure used in (Lv et al., 2020). Specially, any irrelevant label can be chosen as a candidate label with the probability ρ uniformly. When encountering a special case that no one irrelevant label is chosen, we randomly pick one irrelevant label as the candidate label to ensure all training set are fully corrupted.

Comparison Methods We compare the proposed ML-PLL method with the following deep neural network based PLL methods: PiCO (Wang et al., 2022), a contrastive learning based PLL method which disambiguates the candidate labels by the refined pseudo-label in moving-average manner. LWS (Wen et al., 2021), a discriminative PLL method which proposes to trade-off the losses by mean of weighting the risk function on the candidate labels and non-candidate ones; CC (Feng et al., 2020b), a classifier-consistent method which leverages transition matrix to induce an empirical risk estimator; MSE and EXP (Feng et al., 2020a) are two baselines that exploit mean square error and exponential loss as risks for inducing the unbiased risk estimator.

For each comparison methods, we configure it with the same backbone structure for feature extraction and adopt the same predictive model for prediction. Moreover, all the comparison methods are configured with the suggested parameters according to the respective literatures. For all the experiments, we report average test accuracy and standard deviation based on 5 independent runs.

Implementation Details We use LeNet-5 as backbone on Fashion-MNIST and Kuzushiji-MNIST datasets, and 18-layer ResNet as backbone on CIFAR-10 and CIFAR-100 for feature extraction. The projection network is a two-layer MLP that outputs 128-dimensional embeddings. The prediction network is a two-layer MLP, while the Leaky ReLu activation function is used in the middle layer and a softmax activation is adopted in the output layer. The weighted combination parameter λ in Eq.(1), temperature parameter τ in Eq.(2), momentum coefficient in Eq.(5) and momentum smooth coefficient in Eq.(9) are set to 0.99, 0.07, 0.999 and 0.99 respectively. For all comparison experiments, we employ a standard SGD optimizer with a momentum of 0.9, a weight decay of 1e-3 for model training. The mini-batch size, learning rate and total epochs are set to 128, 0.01 and 400

Table 3: Accuracy (mean±std) comparisons on Fashion-MNIST, Kuzushiji-MNIST and CIFAR-10 with uniform noise labels on different ambiguity levels. The best result is highlighted in bold.

Data set	Method	$\rho = 0.1$	$\rho = 0.3$	$\rho = 0.5$
Fashion-MNIST	ML-PLL	92.75±0.05%	92.23±0.12%	91.86±0.08%
	PiCO	**93.30±0.09%**	**93.11±0.08%**	**92.78±0.03%**
	LWS	91.44±0.13%	91.85±0.14%	90.59±0.18%
	CC	92.26±0.12%	91.75±0.04%	90.92±0.06%
	MSE	87.25±0.04%	87.02±0.04%	85.76±0.08%
	EXP	91.02±0.06%	89.88±0.06%	89.21±0.11%
Kuzushiji-MNIST	ML-PLL	**97.63**±0.07%	**97.45±0.05%**	96.69±0.05%
	PiCO	97.58±0.06%	97.32±0.07%	**96.95±0.03%**
	LWS	96.22±0.10%	96.15±0.24%	95.43±0.02%
	CC	96.45±0.04%	96.16±0.02%	95.62±0.10%
	MSE	82.87±0.20%	81.20±0.23%	78.43±0.28%
	EXP	94.98±0.03%	94.21±0.05%	93.56±0.11%
CIFAR-10	ML-PLL	**94.64±0.11%**	94.05±0.15%	92.60±0.21%
	PiCO	94.39±0.18%	**94.18±0.12%**	**93.58±0.11%**
	LWS	90.30±0.60%	88.99±1.43%	86.16±0.85%
	CC	82.30±0.21%	79.08±0.07%	74.05±0.35%
	MSE	79.97±0.45%	75.64±0.28%	67.09±0.66%
	EXP	79.23±0.10%	75.79±0.21%	70.34±1.32%

Table 4: Accuracy (mean±std) comparisons on CIFAR-100 with uniform noise labels on different ambiguity levels. The best result among each column is highlighted in bold.

Data set	Method	$\rho = 0.01$	$\rho = 0.05$	$\rho = 0.1$
CIFAR-100	ML-PLL	**73.45±0.15%**	**73.06±0.13%**	**72.91±0.15%**
	PiCO	73.09±0.34%	72.74±0.30%	69.91±0.11%
	LWS	65.78±0.02%	59.56±0.33%	53.53±0.08%
	CC	49.76±0.45%	47.62±0.08%	35.72±0.47%
	MSE	47.45±1.50%	45.05±1.40%	33.27±2.81%
	EXP	49.17±0.05%	46.02±1.82%	38.81±0.49%

respectively. The parameter α and β are chosen from $\{0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and $\{1, 2, 3, 4, 5, 6\}$ respectively according to the accuracy on a validation dataset (10% of the training dataset).

4.2 Experimental Results on Competitive Label Noise

We compared the proposed ML-PLL method with multiple state-of-art comparison methods on four widely used benchmark datasets corrupted by the competitive label noise and uniform label noise respectively. The comparison results on Fashion-MNIST, Kuzushiji-MNIST, and CIFAR-10 with competitive noise labels is reported in Table 1, from which we can see that the proposed method consistently outperforms all the comparison methods, and the performance gains yield by ML-PLL over all the other methods are quite notable in many cases. For example, the proposed method outperforms the best alternative comparison method by 1.48%, and 2.83% where Avg.#CL = 4, 5 on CIFAR-10 respectively. It is worth noting that with the increase of the average number of candidate labels, consequently boosts the ambiguity level, the performance gains become larger. This clearly demonstrates that the effectiveness of the proposed method in addressing the competitive label noise.

In addition, we further challenge a complicated dataset CIFAR-100 with large label space ($L = 100$). The experimental results are reported in Table 2, from which we can see that the proposed method still outperforms all the comparison methods and has a remarkable performance improvement even on such a complicated dataset. For example, we improve upon the best comparison method by 1.53%, 3.49%, and 4.31% where Avg.#CL = 2, 6, 10 respectively. Moreover, the comparison methods like PiCO and LWS also perform well on the lower ambiguity levels, while the performance drops dramatically with the increase of the ambiguity level. By contrast, the performance of the proposed ML-PLL method consistently outperforms the other algorithms with a notable performance gain. This again validates the efficacy of ML-PLL in addressing the competitive label noise.

4.3 Experimental Results on Uniform Label Noise

To further validate the effectiveness of the proposed method, we also perform experiments on Fashion-MNIST, Kuzushiji-MNIST, CIFAR-10 and CIFAR-100 corrupted by the uniform label noise. All the comparison results are reported in Table 3 and Table 4. As shown in Table 3, we

Table 5: Ablation study on Fashion-MNIST, Kuzushiji-MNIST, CIFAR-10 with Avg.#CL = 4, and CIFAR-100 with Avg.#CL = 6.

Ablation variant	Fashion-MNIST (Avg.#CL = 4)	Kuzushiji-MNIST (Avg.#CL = 4)	CIFAR-10 (Avg.#CL = 4)	CIFAR-100 (Avg.#CL = 6)
Full Model	89.94%	95.38%	83.52%	57.56%
CLS-w/o-ml	86.49%	92.18%	80.17%	55.27%
CLS-w/o-con	84.93%	91.41%	79.08%	54.15%
CLS-w/o-lc	79.94%	82.10%	51.64%	37.51%
CLS	75.90%	77.66%	47.06%	33.60%

can see that although the proposed method is not specially designed for uniform label noise, it still achieves the competitive performance on the whole datasets. Table 4 shows that the superiority of our proposed method is more significant compared with the other methods on such a complicated dataset; moreover, the performance gains are getting larger with the increase of the ambiguity level.

For the complicated dataset CIFAR-100 with large label space, no matter in competitive label noise or uniform label noise learning scenario, the performance gains become larger with ambiguity level increases, i.e., the average number of candidate labels increase. All these results demonstrate the efficacy of the proposed method in handing competitive label noise and uniform label noise for PLL.

4.4 Ablation Study

The objective of the proposed method contains three parts that contribute to PLL: the classification loss, the interactive mutual learning loss and the consistency loss. In addition, noise label correction is an indispensable component in the proposed mutual learning model. To validate each part's importance, we perform an ablation study to compare the proposed method with following ablation variants: 1) CLS-w/o-ml, which drops mutual learning loss; 2) CLS-w/o-con, which drops consistency loss; 3) CLS-w/o-lc, which drop noise label correction procedure; 4) CLS, which only uses classification loss by dropping mutual learning loss, consistency loss and noise label correction procedure. The comparison results are reported in Table 5. From Table 5, we can see that compared with the full model, all four variants produced inferior results, which demonstrates that all the components contribute to the full model in different degree. Among the three variants, CLS-w/o-ml, CLS-w/o-con and CLS-w/o-lc, we can see that CLS-w/o-lc, which drops the noise label correction procedure, produces a large performance degradation. This makes sense since dropping noise label correction means giving up correcting the competitive label noise, the prediction network based classifier has to be induced on the competitive noise labels and models a noise label distribution rather than the true label distribution. Among the two variants, CLS-w/o-ml and CLS-w/o-con, we can see that both of them produce a performance degradation to some extent, since CLS-w/o-ml drops the mutual learning loss that destroys the mutual learning framework, and CLS-w/o-con drops the consistency loss that makes the proposed model can only address the uniform label noise while powerless to the competitive label noise. Moreover, CLS-w/o-con produces more inferior performance than CLS-w/o-ml in most cases, which demonstrates the efficacy of the proposed transformation network in exploring the association relationships between the competitive noise labels and the ground-truth label. Overall, these ablation results demonstrates that the proposed ML-PLL method is effective by integrating these parts for addressing partial label learning with competitive label noise.

5 Conclusion

This paper considers a more realistic PLL scenario with competitive label noise which can be more difficult to distinguish with the ground-truth label than the random noise labels. We propose a novel mutual partial label learning approach called ML-PLL to address this challenging problem. ML-PLL learns a prediction network based classifier and a class-prototype based classifier cooperatively with label correction under the mutual learning framework. Moreover, a transformation network is introduced to model the association relationships between the ground-truth label and candidate noise labels, and learn it together with the prediction network to match the observed candidate labels and enhance label correction. The comprehensive experiments on several benchmark PLL datasets demonstrate that the proposed approach significantly outperforms the state-of-art PLL approaches.

REFERENCES

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

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

Lei Feng and Bo An. Leveraging latent label distributions for partial label learning. In *Twenty-fourth international joint conference on artificial intelligence (IJCAI)*, pp. 2107–2113, 2018.

Lei Feng and Bo An. Partial label learning with self-guided retraining. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3542–3549, 2019.

Lei Feng, Takuo Kaneko, Bo Han, Gang Niu, Bo An, and Masashi Sugiyama. Learning with multiple complementary labels. In *International Conference on Machine Learning (ICML)*, pp. 3072–3081. PMLR, 2020a.

Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. Provably consistent partial-label learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:10948–10960, 2020b.

Lei Feng, Senlin Shu, Zhuoyi Lin, Fengmao Lv, Li Li, and Bo An. Can cross entropy loss be robust to label noise? In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (IJCAI)*, pp. 2206–2212, 2021.

Chen Gong, Tongliang Liu, Yuanyan Tang, Jian Yang, Jie Yang, and Dacheng Tao. A regularization approach for instance-based superset label learning. *IEEE transactions on cybernetics*, 48(3): 967–978, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 9729–9738, 2020.

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

Rong Jin and Zoubin Ghahramani. Learning with multiple labels. *Advances in neural information processing systems (NeurIPS)*, 15, 2002.

Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. *Advances in neural information processing systems (NeurIPS)*, 30, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Junnan Li, Caiming Xiong, and Steven CH Hoi. Mopro: Webly supervised learning with momentum prototypes. *arXiv preprint arXiv:2009.07995*, 2020.

Liping Liu and Thomas Dietterich. A conditional multinomial mixture model for superset label learning. *Advances in neural information processing systems (NeurIPS)*, 25, 2012.

Jie Luo and Francesco Orabona. Learning from candidate labeling sets. *Advances in neural information processing systems (NeurIPS)*, 23, 2010.

Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, and Masashi Sugiyama. Progressive identification of true labels for partial-label learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 6500–6510. PMLR, 2020.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems (NeurIPS)*, 26, 2013.

Nam Nguyen and Rich Caruana. Classification with partial labels. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pp. 551–559, 2008.

Yassine Ouali, Céline Hudelot, and Myriam Tami. An overview of deep semi-supervised learning. *arXiv preprint arXiv:2006.05278*, 2020.

Senlin Shu, Zhuoyi Lin, Yan Yan, and Li Li. Learning from multi-class positive and unlabeled data. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 1256–1261. IEEE, 2020.

Cai-Zhi Tang and Min-Ling Zhang. Confidence-rated discriminative partial label learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.

Deng-Bao Wang, Min-Ling Zhang, and Li Li. Adaptive graph guided disambiguation for partial label learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. Pico: Contrastive label disambiguation for partial label learning. *arXiv preprint arXiv:2201.08984*, 2022.

Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. Leveraged weighted loss for partial label learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 11091–11100. PMLR, 2021.

Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. Revisiting consistency regularization for deep partial label learning. In *International Conference on Machine Learning (ICML)*, pp. 24212–24225, 2022.

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

Ming-Kun Xie and Sheng-Jun Huang. Partial multi-label learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 946—-952, 2018.

Ning Xu, Jiaqi Lv, and Xin Geng. Partial label learning via label enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 5557–5564, 2019.

Ning Xu, Congyu Qiao, Xin Geng, and Min-Ling Zhang. Instance-dependent partial label learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

Yan Yan and Yuhong Guo. Adversarial partial multi-label learning with label disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 10568–10576, 2021.

Yao Yao, Jiehui Deng, Xiuhua Chen, Chen Gong, Jianxin Wu, and Jian Yang. Deep discriminative cnn with temporal ensembling for ambiguously-labeled image classification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 12669–12676, 2020a.

Yao Yao, Chen Gong, Jiehui Deng, and Jian Yang. Network cooperation with progressive disambiguation for partial label learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECCV)*, pp. 471–488. Springer, 2020b.

Fei Yu and Min-Ling Zhang. Maximum margin partial label learning. In *Asian conference on machine learning (ACML)*, pp. 96–111. PMLR, 2016.

Zinan Zeng, Shijie Xiao, Kui Jia, Tsung-Han Chan, Shenghua Gao, Dong Xu, and Yi Ma. Learning by associating ambiguously labeled images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 708–715, 2013.

Min-Ling Zhang and Fei Yu. Solving the partial label learning problem: An instance-based approach. In *Twenty-fourth international joint conference on artificial intelligence (IJCAI)*, 2015.

Min-Ling Zhang, Bin-Bin Zhou, and Xu-Ying Liu. Partial label learning via feature-aware disambiguation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1335–1344, 2016.

Min-Ling Zhang, Fei Yu, and Cai-Zhi Tang. Disambiguation-free partial label learning. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2155–2167, 2017.

Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1): 44–53, 2018.

A APPENDIX

In this appendix, we provide the complete implementation of our proposed approach in subsection A.1. Moreover, the results of parameter sensitivity analysis is reported in subsection A.2, which studies how the trade-off parameters impact performance of the proposed model. Moreover, subsection A.3 conducts extended experiments on the CIFAR-10 dataset to explore how would the noise level affect performance of the proposed ML-PLL method. The characteristics of the dataset is recorded in A.4. In addition, an additional experiment is conducted to further validate the effectiveness of the proposed ML-PLL method in subsection A.5.

A.1 PSEUDO-CODE OF ML-PLL

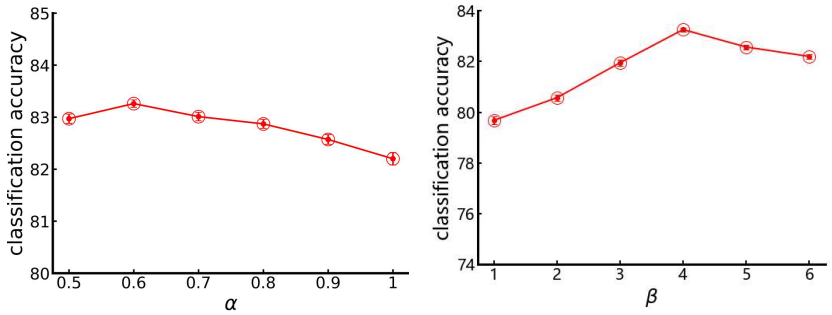
We summarize the pseudo-code of our proposed ML-PLL method in Algorithm 1.

Algorithm 1 Training Algorithm of ML-PLL.

Input:
D: the PLL training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$. F: prediction network based classifier. G: feature extractor. H: projection network. α, β: trade-off hyperparameters.
for number of training iterations **do**
  Sample a batch B with m samples from D.
  // Noise label correction
  Get the class prototype-based prediction probability vector $\mathbf{p}_i$ via Eq.(2).
  Conduct noise label correction via Eq.(1).
  Get normalized soft pseudo-label $\widehat{q}_i$ via Eq.(3).
  // Batch-wise moving average based prototype update
  Compute the class prototype via Eq.(4).
  Update the class prototype via Eq.(5).
  // Update of the transformation matrix
  Compute and update the transformation matrix via Eq.(9).
  // Classification loss calculation
  $\mathcal{L}_c(F, G, H) = \frac{1}{m} \sum_{\mathbf{x}_i \sim B} \sum_{k=1}^{L} -\widehat{q}_{ik} \log(F^k(G(\mathbf{x}_i)))$
  // Interactive mutual learning loss calculation
  $\mathcal{L}_{ml}(F, G, H) = \frac{1}{m} \sum_{\mathbf{x}_i \sim B} \sum_{k=1}^{L} -F^k(G(\mathbf{x}_i)) \log(p_{ik})$
  // Consistency loss calculation
  $\mathcal{L}_{con}(F, G, H, T) = \sum_{\mathbf{x}_i \sim B} \Big(\mathrm{KL}\big(F(G(\mathbf{x}_i))R, \widetilde{Y}_i\big) + \mathrm{KL}\big(\widetilde{Y}_i, F(G(\mathbf{x}_i))R\big) \Big)$
  // Network updating
  $\mathcal{L} = \mathcal{L}_c + \alpha \mathcal{L}_{ml} + \beta \mathcal{L}_{con}$
  Update network F, T, G and H to minimize $\mathcal{L}$.
end for

A.2 PARAMETER SENSITIVITY ANALYSIS

In this subsection, we study the trade-off parameters α and β which controls the relative importance of the mutual learning loss and consistency loss respectively, and perform the experiments on CIFAR-10 dataset to study how would the trade-off parameters α and β affect the performance of ML-PLL. The experimental ranges for the parameters α and β are chosen from



(a) Vary the parameter α (b) Vary the parameter β

Figure 2: Test performance with different α and β values for the proposed ML-PLL approach on the CIFAR-10 dataset.

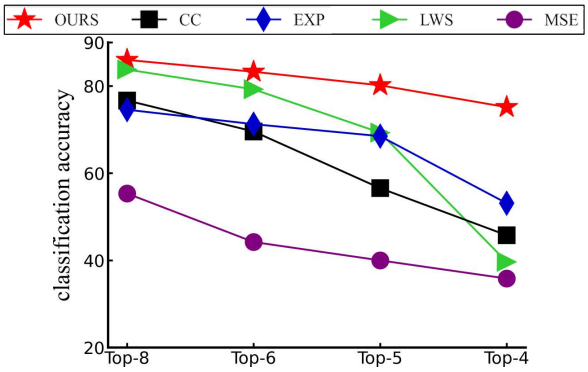


Figure 3: Impact of Competitive Noise Level for ML-PLL on the CIFAR-10 dataset.

$\{0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and $\{1, 2, 3, 4, 5, 6\}$ respectively. Note that a larger α means higher weight of the mutual learning loss; while a larger β means higher importance of the consistency loss. The experimental results with different parameter of α and β on the corrupted CIFAR-10 dataset with Avg#CL = 4 are reported in Figure 2(a) and Figure 2(b) respectively.

We vary β while keep α with its best. As shown in Figure 2(b), we can see that when β is very small, the performance is relatively poor. With the increase of β, the performance improves as the consistency loss begin to work. This suggests that the consistency loss is an important loss term for the proposed model. When β is too large, the performance degrades since the consistency loss dominates. This make sense since we just hope the consistency loss to assist the model training, rather than dominates. We can also observe the similar performance changes by varying the value of the trade-off parameter α.

A.3 Impact of Competitive Noise Levels

In subsections 4.2, the experimental results has shown the effectiveness of the proposed method by enhancing the competitive level through increasing number of the average candidate labels. In this subsection, we further validate its effectiveness by increasing the competitive level in another perspective: decreasing the competitive label space. We conduct an extension experiment to demonstrate how the performance of PLL algorithms changes on the different competitive level, the comparison results according to the accuracy on the corrupted CIFAR-10 dataset with Avg#CL = 4 are reported in Figure 3. Note that k is the number of competitive label space, we can choose the candidate labels from the competitive label set. With the decrease of the competitive label space k, the competitive level increases since the noisy labels from the smaller competitive label space are more difficult for label disambiguation. As shown in Figure 3, we can see that with the decrease of the k, consequently increase the competitive level, the performance of all the comparison meth-

Table 6: Characteristic of the datasets corrupted by the competitive noise label generation process.

Dataset	#Train	#Test	#Feature	Classes	Avg.#CL
Fashion-MNIST	60,000	10,000	784	10	3,4,5
Kuzushiji-MNIST	60,000	10,000	784	10	3,4,5
CIFAR-10	50,000	10,000	3,072	10	3,4,5
CIFAR-100	50,000	10,000	3,072	100	2,6,10

Table 7: Accuracy (mean±std) on CIFAR-10 with different noise label generation process.

Method	ML-PLL	LWS	CC	EXP	MSE
Case (1)	**94.91±0.04%**	90.82±0.03%	90.57±0.11%	76.43±0.08%	65.83±0.19%
Case (2)	**93.11±0.08%**	69.42±0.07%	69.53±0.12%	60.47±0.14%	40.03±0.20%

ods degrades in general. However, the proposed ML-PLL method consistently outperforms all the other four comparison methods. It is worth noting that the performance of ML-PLL actually is quite stable with the increase of the competitive level (the decrease of the k), and the performance gain becomes larger when the other methods encounter dramatically performance degradation. This again validates the effectiveness of ML-PLL in addressing competitive label noise.

A.4 DATASET

The average number of candidate labels (Avg.#CL) and the characteristics of each benchmark dataset corrupted by the competitive label noise generation process is recorded in Table 6.

A.5 THE INFLUENCE OF NOISE LABEL GENERATION

To further explore the impact of noise label generation on model performance, we follow (Wen et al., 2021) to conduct additional experiment with following setting: some irrelevant labels that have larger association relationships than other labels would have larger probability of label flipping ρ than other labels. We perform the experiments on CIFAR-10 with following commonly used learning cases:

$$(1) = \begin{bmatrix} 1 & 0.5 & 0 & \cdots & 0 \\ 0 & 1 & 0.5 & \cdots & 0 \\ \vdots & & & \cdots & \vdots \\ 0.5 & 0 & 0 & \cdots & 1 \end{bmatrix}, \qquad (2) = \begin{bmatrix} 1 & 0.9 & 0.7 & 0.5 & 0.3 & 0.1 & 0 & \cdots & 0 \\ 0 & 1 & 0.9 & 0.7 & 0.5 & 0.3 & 0.1 & \cdots & 0 \\ \vdots & & & & \cdots & & & & \vdots \\ 0.9 & 0.7 & 0.5 & 0.3 & 0.1 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

From the two flipping matrices we can see that each entry denotes the probability of label being a candidate label coupled with the ground-truth label. The comparison results is reported in Table 7, from which we can see that: in a simple case (1), the proposed ML-PLL method outperforms the best comparison methods with a slight advantages; it is worth noting that in case (2), ML-PLL method still achieves a competitive performance and has a remarkable performance improvement even in such a harder case. These additional experimental results further demonstrates the effectiveness of the proposed method in candidate label disambiguation.