Appendix

CONTENTS

A NOTATION

We present the notation table for each symbol used in this paper in Table 6.

B RELATED WORK

Many previous methods have been proposed for dealing with the specific types and some combinations of imprecise label configurations. In the main paper, we only introduced the representative pipelines for partial label learning, semi-supervised learning, and noisy label learning, as shown in Section 2. In this section, we discuss in more detail the previous methods. We also extend discussions about the imprecise label settings to multi-instance learning and mixed imprecise label learning.

B.1 PARTIAL LABEL LEARNING

Label ambiguity remains a fundamental challenge in Partial Label Learning (PLL). The most straightforward approach to handling PLL is the average-based method, which assumes an equal probability for each candidate label being the ground-truth label. For instance, Hüllermeier & Beringer (2006) employed k-nearest neighbors for label disambiguation, treating all candidate labels of a sample's neighborhood equally and predicting the ground-truth label through voting strategies. However, a significant drawback of average-based methods is that false positive labels can mislead them.

To overcome these limitations, researchers have explored identification-based methods for PLL. In contrast to average-based methods, which treat all candidate labels equally, identification-based

Table 6: Notation Table

Notation	Definition
$\mathbf{x}$	A training instance
y	A class index label
$[\iota]$	An imprecise label, which might contain multiple class indices
$X = \{\mathbf{x}_i\}_{i\in[N]}$	Data. A set of data instances $\mathbf{x}$ of size N
$\mathcal{X}$	Input space where $\mathbf{x}$ is drawn from
$Y = \{y_i\}_{i\in[N]}$	Ground-truth labels. A set of label indices y of size N
$\mathcal{Y}$	Label space where y is drawn from
$I = \{[\iota]_i\}_{i\in[N]}$	Imprecise labels. A set of imprecise labels $[\iota]$ of size N
f	Model backbone
g	Model classifier
h	Model multi-layer perceptron
$f \circ g$	Model mapping $\mathcal{X} \to \mathcal{Y}$
θ	Learnable parameters of $f \circ g$
$\mathbf{p}(y\|\mathbf{x};\theta)$	Output probability from model $f \circ g$
$f \circ h$	Model mapping $\mathcal{X} \to \mathcal{Z}$, where Z is a projected feature space
$\mathcal{D}$	Dataset
$\mathcal{L}$	Loss function
$\mathcal{A}_\mathrm{w}$	Weak data augmentation, usually is HorizontalFlip
$\mathcal{A}_\mathrm{s}$	Strong data augmentation, usually is RandAugment (Cubuk et al., 2020)
$\mathbf{z}_\mathrm{w}$	Projected features from $f \circ h$ on weakly-augmented data
$\mathbf{z}_\mathrm{s}$	Projected features from $f \circ h$ on strongly-augmented data
$\mathcal{M}$	Memory queue in MoCo (He et al., 2020)
$\mathbf{s}$	A partial label, with ground-truth label contained
S	A set of partial labels
$\mathbf{x}^\mathrm{l}$	A labeled training example
y^l	A labeled class index
$\mathbf{x}^\mathrm{u}$	A unlabeled training example
y^u	A unknown class index for unlabeled data
X^L	A set of labeled data instances
Y^L	A set of labels for labeled data instances
X^U	A set of unlabeled data instances
Y^U	A set of unknown labels for unlabeled data instances
$\hat{p}^\mathbf{u}$	The maximum predicted probability on unlabeled data $\max(\mathbf{p}(y\|\mathbf{x}^\mathrm{u};\theta))$
$\hat{y}^\mathrm{u}$	The pseudo-label from the predicted probability on unlabeled data $\arg\max(\mathbf{p}(y\|\mathbf{x}^\mathrm{u};\theta))$
τ	The threshold for confidence thresholding
$\hat{y}$	A corrupted/noisy label
$\hat{y}^\mathrm{oh}$	An one-hot version of the corrupted/noisy label
$\hat{Y}$	A set of noisy labels
$\mathbf{u}, \mathbf{v}, \mathbf{m}$	Noise model related parameters in SOP (Liu et al., 2022)
$\alpha(i, y)$	The forward score in forward-backward algorithm
$\beta(i, y)$	The backward score in forward-backward algorithm
$\mathcal{T}(\hat{y}\|y;\omega)$	The simplified noise transition model in ILL
ω	The parameters in the simplified noise model

methods view the ground-truth label as a latent variable. They seek to maximize its estimated probability using either the maximum margin criterion (Nguyen & Caruana, 2008; Zhang et al., 2016b) or the maximum likelihood criterion (Liu & Dietterich, 2012). Deep learning techniques have recently been incorporated into identification-based methods, yielding promising results across multiple datasets. For example, PRODEN (Lv et al., 2020) proposed a self-training strategy that disambiguates candidate labels using model outputs. CC (Feng et al., 2020b) introduced classifier-consistent and risk-consistent algorithms, assuming uniform candidate label generation. LWS (Wen et al., 2021) relaxed this assumption and proposed a family of loss functions for label disambiguation. More recently, Wang et al. (2022a) incorporated contrastive learning into PLL, enabling the model to learn discriminative representations and show promising results under various levels of ambiguity.

Nevertheless, these methods rely on the fundamental assumption that the ground-truth label is present in the candidate set. This assumption may not always hold, especially in cases of unprofessional judgments by annotators, which could limit the applicability of these methods in real-world scenarios.

B.2 Semi-Supervised Learning

Consistency regularization and self-training, inspired by clusterness and smoothness assumptions, have been proposed to encourage the network to generate similar predictions for inputs under varying

perturbations (Tarvainen & Valpola, 2017; Samuli & Timo, 2017; Miyato et al., 2018). Self-training (Lee et al., 2013; Arazo et al., 2020; Sohn et al., 2020) is a widely-used approach for leveraging unlabeled data. Pseudo Label (Lee et al., 2013), a well-known self-training technique, iteratively creates pseudo labels that are then used within the same model. However, this approach suffers from confirmation bias (Arazo et al., 2020), where the model struggles to rectify its own errors when learning from inaccurate pseudo labels. Recent studies focus largely on generating high-quality pseudo-labels. MixMatch (Berthelot et al., 2019b), for instance, generates pseudo labels by averaging predictions from multiple augmentations. Other methods like ReMixMatch (Berthelot et al., 2019a), UDA (Xie et al., 2020a), and FixMatch (Sohn et al., 2020) adopt confidence thresholds to generate pseudo labels for weakly augmented samples, which are then used to annotate strongly augmented samples. Methods such as Dash (Xu et al., 2021), FlexMatch (Zhang et al., 2021b), and FreeMatch (Wang et al., 2023) dynamically adjust these thresholds following a curriculum learning approach. SoftMatch (Chen et al., 2023) introduces a novel utilization of pseudo-labels through Gaussian re-weighting. Label Propagation methods (Iscen et al., 2019) assign pseudo labels based on the local neighborhood's density. Meta Pseudo Labels (Pham et al., 2021) proposes generating pseudo labels with a meta learner. SSL learning has also seen improvements through the incorporation of contrastive loss (Li et al., 2021a; Zheng et al., 2022). Furthermore, MixUp (Zhang et al., 2017) has shown its effectiveness in a semi-supervised learning (SSL) (Berthelot et al., 2019b;a; Cai et al., 2022).

B.3 Noisy Label Learning

Due to their large number of parameters, deep neural networks are prone to overfitting to noisy labels. Although certain popular regularization techniques like mixup (Zhang et al., 2017) can somewhat mitigate overfitting, they fall short of completely resolving the issue of learning with noisy labels, as they lack accurate noise modeling. Numerous techniques have been proposed to handle Noisy Label Learning (NLL) (Song et al., 2022). They can broadly be divided into three categories: robust loss functions (Ghosh et al., 2017; Zhang & Sabuncu, 2018; Wang et al., 2019c; Ma et al., 2020b), noise estimation (Xiao et al., 2015a; Goldberger & Ben-Reuven, 2016; Liu et al., 2020; Zhang et al., 2021d; Northcutt et al., 2021), and noise correction (Han et al., 2018; Li et al., 2020; Liu et al., 2022).

Designing loss functions that are robust to noise is a well-explored strategy for tackling the label noise problem. The l_1 loss [59], which is a robust loss function, is a popular choice and has seen many recent extensions (Zhang & Sabuncu, 2018; Wang et al., 2019c; Ma et al., 2020a; Yu et al., 2020). Additionally, methods that re-weight loss (Liu & Tao, 2016) have also been explored for learning with noisy labels. However, despite enabling the model to learn faster from accurate labels, these robust loss functions still lead to overfitting to corrupted labels when used with models having a large number of parameters.

Another common strategy to handle label noise involves assuming that the noisy label originates from a probability distribution that depends on the actual label. The key task here is to estimate the underlying transition probabilities. Early works (Goldberger & Ben-Reuven, 2016) incorporated these transition probabilities into a noise adaptation layer that is stacked over a classification network and trained in an end-to-end fashion. More recent work, such as Forward (Patrini et al., 2016), prefers to estimate these transition probabilities using separate procedures. However, the success of this method is contingent upon the availability of clean validation data (Northcutt et al., 2021) or additional assumptions about the data (Zhang et al., 2021e).

Noise correction has shown promising results in noisy label learning recently. During the early learning phase, the model can accurately predict a subset of the mislabeled examples (Liu et al., 2020). This observation suggests a potential strategy of correcting the corresponding labels. This could be accomplished by generating new labels equivalent to soft or hard pseudo-labels estimated by the model (Tanaka et al., 2018; Yi & Wu, 2019). Co-Teaching uses multiple differently trained networks for correcting noisy labels (Han et al., 2018). SELFIE (Song et al., 2019) corrects a subset of labels by replacing them based on past model outputs. Another study in Arazo et al. (2019) uses a two-component mixture model for sample selection, and then corrects labels using a convex combination. They also utilize mixup (Zhang et al., 2017) to enhance performance. Similarly, DivideMix (Li et al., 2020) employs two networks for sample selection using a two-component mixture model and implements the semi-supervised learning technique MixMatch (Berthelot et al., 2019b).

B.4 MULTI-INSTANCE LEARNING

Multiple Instance Learning (MIL) is a sub-field of supervised learning where each instance is associated with a label. However, MIL stands out due to its handling of incomplete label knowledge in the training set. The training data in MIL is composed of 'bags', each containing several unlabeled instances. The primary objective of MIL is to predict labels for new, unseen bags. Dietterich et al. (1997) pioneered the concept of MIL, specifically applying it to drug activity prediction. This innovative approach has since inspired the development of numerous algorithms tailored to tackle this distinctive problem. One pragmatic approach for dealing with label noise in this context involves counting the number of positive instances within a bag and setting a threshold for classifying positive bags. This technique was encapsulated as the threshold-based assumption for MIL in (Foulds & Frank, 2010). Per this assumption, a bag is classified as positive if and only if the count of positive instances exceeds a predefined threshold. This constituted the initial integration of counting information into the MIL framework. Following this development, various efforts (Tao et al., 2004a;b) focused on carrying out bag-level predictions based on count-based assumptions.

Another significant trajectory in MIL research is centered on instance-level prediction. The framework proposed by Maron & Lozano-Pérez (1997) is particularly renowned for instance-level prediction within the MIL paradigm. This framework has underpinned numerous research proposals that have demonstrated effectiveness. The foundational idea behind these frameworks is the dynamic or static labeling of instances according to the bag label. Empirical studies, as detailed in (Vanwinckelen et al., 2016), have illustrated that superior bag-level prediction does not necessarily guarantee improved instance-level prediction. To further the field of MIL, addressing these critical issues will be essential in order to develop more robust, flexible, and scalable MIL methods that can be applied across a broad range of real-world situations (Zhang et al., 2021a).

B.5 MIXED IMPRECISE LABEL LEARNING

Various previous works have explored dealing with distinct types of imprecise labels. However, they have yet to tackle a combination of partial labels, limited labels, and noisy labels, which is a highly realistic scenario. For instance, recent attention has been paid to the issue of partial noisy label learning.

PiCO+ (Wang et al., 2022b), an extended version of PiCO (Wang et al., 2022a), is tailored specifically for partial noisy labels. It employs a distance-based method to select clean samples and uses a semi-supervised contrastive learning algorithm to train robust classifiers. This approach distinguishes between clean and noisy samples and enhances the learning of distinctive features. IRNet (Lian et al., 2022b) is a novel framework designed for Partial Label Learning with noisy labels. It uses two modules: noisy sample detection and label correction, transforming the scenario of noisy PLL into a more traditional PLL. DALI (Xu et al., 2023) is another framework designed to reduce the negative impact of detection errors by creating a balance between the initial candidate set and model outputs, with theoretical assurances of its effectiveness.

Additionally, some work has focused on semi-supervised partial label learning (Wang et al., 2019b; Wang & Zhang, 2020). An iterative label propagation process is proposed in these studies, operating between partially labeled examples and unlabeled instances, which helps clarify the candidate label sets of the partially labeled examples and assign valid labels to unlabeled instances. However, these methods don't scale well to larger datasets like CIFAR-10.

Our study and the proposed framework hold significant importance as we have not found any existing research that can effectively address the challenge of handling a combination of partial, limited, and noisy labels simultaneously. This underscores the novelty and significance of our work, making it a valuable contribution to the field of machine learning.

(a) Full Label (b) Partial Label (c) Semi-Supervised (d) Noisy Label

Figure 3: Illustration of NFA for different imprecise label configurations on 4 instances with 3 classes. We use edge to denote the transition paths for each symbol y and circle to denote the finite state $\mathbf{x}$. Different label configurations I correspond to set the transition paths in the NFA, where we use red solid edge for available paths and blue dash edge for unavailable paths. (a) Full label, the transition paths are definite Y. (b) Partial label, the transition paths follow candidate symbols S. (c) Semi-supervised, all paths are available for unlabeled data. (d) Noisy label; all paths are available.

C METHODS

C.1 DERIVATION OF VARIATIONAL LOWER BOUND

Evidence lower bound (ELBO), or equivalently variational lower bound (Dempster et al., 1977), is the core quantity in EM. From Eq. (5), to model $\log P(X, I; \theta)$, we have:

$$\begin{aligned}
\log P(X, I; \theta) &= \int Q(Y) \log P(X, I; \theta) dY \\
&= \int Q(Y) \log P(X, I; \theta) \frac{P(Y|X, I; \theta)}{P(Y|X, I; \theta)} dY \\
&= \int Q(Y) \log \frac{P(X, I, Y; \theta) Q(Y)}{P(Y|X, I; \theta) Q(Y)} dY \\
&= \int Q(Y) \log \frac{P(X, I, Y; \theta)}{Q(Y)} dY - \int Q(Y) \log \frac{P(Y|X, I; \theta)}{Q(Y)} dY
\end{aligned} \tag{10}$$

where the first term is the ELBO and the second term is the KL divergence $\mathcal{D}_{KL}(Q(Y)||P(Y|X, I; \theta))$. Replacing $Q(Y)$ with $P(Y|X, I; \theta^t)$ at each iteration will obtain Eq. (5).

C.2 INSTANTIATIONS MIXED IMPRECISE LABEL LEARNING

In this setting, we have both labeled data and unlabeled data, where the labels for the labeled data are both partial and noisy. On the unlabeled data, the unsupervised objective is the same as the unsupervised consistency regularization of semi-supervised learning shown in Eq. (7). On the labeled data, it mainly follows the Eq. (9) of noisy label learning, with the noisy single label becoming the noisy partial labels $\hat{\mathbf{s}}$. For noisy partial labels, the noisy supervised objective in Eq. 8 becomes the supervised consistency regularization as in Eq. 6 of partial label setting to train the noise transition model, and the noisy unsupervised objective becomes the consistency regularization of the prediction conditioned on noisy partial labels:

$$\mathcal{L}_{\text{CE}}\left(\mathbf{p}\left(y \mid \mathcal{A}_{\text{s}}(\mathbf{x}), \hat{\mathbf{s}}; \theta, \omega^t\right), \mathbf{p}\left(y \mid \mathcal{A}_{\text{w}}(\mathbf{x}), \hat{y}; \theta^t, \omega^t\right)\right) + \mathcal{L}_{\text{CE}}\left(\mathbf{p}\left(\hat{y} \mid \mathcal{A}_{\text{w}}(\mathbf{x}); \theta, \omega\right), \hat{\mathbf{s}}\right) \tag{11}$$

We can compute both quantity through the noise transition model:

$$\mathbf{p}(y|\mathbf{x}, \hat{\mathbf{s}}; \theta, \omega^t) \propto \mathbf{p}(y|\mathbf{x}; \theta) \prod_{\hat{y} \in \hat{\mathbf{s}}} \mathcal{T}(y|\hat{y}; \omega^t), \text{and } \mathbf{p}(\hat{y}|\mathbf{x}; \theta, \omega) = \sum_{y \in [C]} \mathbf{p}(y|\mathbf{x}; \theta) \mathcal{T}(\hat{y}|y; \omega). \tag{12}$$

C.3 FORWARD-BACKWARD ALGORITHM ON THE NFA OF IMPRECISE LABELS

This section provides an alternative explanation for the EM framework proposed in the main paper. We treat the problem of assigning labels Y to inputs X as generating a sequence of symbols Y, such that I is satisfied. Such a process can be represented as a finite-state automaton (FSA); more specifically, a non-deterministic finite automaton (NFA) (Hopcroft et al., 2001). The NFA of imprecise labels can be formally defined with a finite set of states X, a finite set of symbols Y, a transition function $f \circ g$, a start state of x_0, and the accepting state of I, determining the available transition paths in the NFA, as shown in Fig. 3. We demonstrate that, Although we can directly derive the

soft-targets utilized in loss functions of each setting we study from Eq. (5), they can also be computed by performing a forward-backward algorithm on the NFA. While there is no difference in the actual value for conducting the forward-backward algorithm and deriving the soft-targets directly, the NFA indeed becomes more necessary when only collection-level imprecise information is provided, where naively considering all possible labeling requires exponential complexity, and forward-backward on NFA reduces the complexity to linear. The reason for the equivalence in the partial, semi-sup, noisy label is because the NFA directly collapses to $\mathcal{O}(1)$ for computing the posterior without any statistical information imposed from I.

To compute the posterior probability $P(Y|X, I; \theta)$, without loss of generality, we assume each training sample is drawn independently and identically, and training samples occur as a permutation-invariant sequence. We further assume that each y_i is conditionally independent given the whole sequence of training samples X, which allows us to re-write Eq. (5) as:

$$\theta^{t+1} = \arg\max_{\theta} \sum_{i\in[N]} \sum_{y\in[C]} P(y_i = y|X, I; \theta^t) \left[\log P(y_i = y|X; \theta) + \log P(I|X, Y; \theta)\right]. \quad (13)$$

The problem of computing the posterior then reduces to calculate the joint probability of I and the latent ground truth label of the i-th sample taking the value y given the whole input sequence X:

$$P(y_i = y|X, I; \theta^t) = \frac{P(y_i = y, I|X; \theta^t)}{P(I|X; \theta^t)}. \quad (14)$$

Subsequently, $P(y_i = y, I|X; \theta^t)$ can be computed in linear time complexity $\mathcal{O}(NC)$ at maximum using the forward-backward algorithm on the NFA, similar to connectionist temporal classification (CTC) (Graves et al., 2006). More specifically, for any symbol y_i in the NFA of imprecise label information, the joint is calculated with the forward score $\alpha(i, y)$ and the backward score $\beta(i, y)$:

$$\begin{aligned} \alpha(i, y) &= \sum_{y'\in y_{i-1}|I} \alpha(i-1, y') P(y_i = y|X; \theta^t), \\ \hat{\beta}(i, y) &= P(y_i = y|X; \theta^t) \sum_{y'\in\in y_{i+1}|I} \beta(i+1, y'), \;\; \beta(i, y) = \frac{\hat{\beta}(i, y)}{P(y_i = y|X; \theta^t)}, \end{aligned} \quad (15)$$

where $\alpha(i, y)$ indicates the total probability of all paths of I at state $\mathbf{x}_i$ and $\beta(i, y)$ indicates the total probability of all paths of I at state $\mathbf{x}_i$. We can then compute the posterior probability:

$$P(y_i = y|X, I; \theta^t) = \frac{\alpha(i, y)\beta(i, y)}{\sum_{y'\in[\iota]_i} \alpha(i, y')\beta(i, y')}, \quad (16)$$

which can be in turn used in Eq. (5). Now the difference in learning with various imprecise label configurations lies only in how to construct the NFA graph for different configurations.

An illustration for the NFA of full labels, partial labels, limited labels with unlabeled data, and noisy labels is shown in Fig. 3. For full precise labels, there is only one determined transition of Y. For partial labels, the transition paths can only follow the label candidates, resulting in the normalized soft-targets with masses on the label candidates as we shown in Section 3.2. For semi-supervised learning, the transition paths for the labeled data strictly follow Y^{L}. For unlabeled data, since there is no constraint on the transition paths, all transitions are possible, corresponding to the soft pseudo-targets. For noisy labels, all transition paths are possible, but the joint probability needs to be computed with the noise transition model. We will show, in future work, the more extensions of the ILL framework on collection-level and statistical imprecise labels, with the help of NFA.

D Experiments

D.1 Additional Training Details

We adopt two additional training strategies for the ILL framework. The first is the "strong-weak" augmentation strategy (Xie et al., 2020a). Since there is a consistency regularization term in each imprecise label formulation of ILL, we use the soft pseudo-targets of the weakly-augmented data to train the strongly-augmented data. The second is the entropy loss (John Bridle, 1991) for class balancing, which is also adopted in SOP (Liu et al., 2022) and FreeMatch (Wang et al., 2023). We set the loss weight for the entropy loss uniformly for all experiments as 0.1.

D.2 PARTIAL LABEL LEARNING

D.2.1 SETUP

Following previous work (Xu et al., 2022; Wen et al., 2021; Wang et al., 2022a), we evaluate our method on partial label learning setting using CIFAR-10, CIFAR-100, and CUB-200 (Welinder et al., 2010). We generate partially labeled datasets by flipping negative labels to false positive labels with a probability q, which is also denoted as a partial ratio. Specifically, the $C-1$ negative labels are uniformly aggregated into the ground truth label to form a set of label candidates. We consider $q \in \{0.1, 0.3, 0.5\}$ for CIFAR-10, $q \in \{0.01, 0.05, 0.1\}$ for CIFAR-100, and $q = 0.05$ for CUB-200. For CIFAR-10 and CIFAR-100, we use ResNet-18 (He et al., 2016) as backbone. We use SGD as an optimizer with a learning rate of 0.01, a momentum of 0.9, and a weight decay of $1\mathrm{e}{-3}$. For CUB-200, we initialize the ResNet-18 (He et al., 2016) with ImageNet-1K (Deng et al., 2009) pre-trained weights. We train 800 epochs for CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), and 300 epochs for CUB-200, with a cosine learning rate scheduler. For CIFAR-10 and CIFAR-100, we use an input image size of 32. For CUB-200, we use an input image size of 224. A batch size of 256 is used for all datasets. The choice of these parameters mainly follows PiCO (Wang et al., 2022a). We present the full hyper-parameters systematically in Table 7.

Table 7: Hyper-parameters for **partial label learning** used in experiments.

Hyper-parameter	CIFAR-10	CIFAR-100	CUB-200
Image Size	32	32	224
Model	ResNet-18	ResNet-18	ResNet-18 (ImageNet-1K Pretrained)
Batch Size	256	256	256
Learning Rate	0.01	0.01	0.01
Weight Decay	1e-3	1e-3	1e-5
LR Scheduler	Cosine	Cosine	Cosine
Training Epochs	800	800	300
Classes	10	100	200

D.2.2 DISCUSSION

We additionally compare our method with R-CR (Wu et al., 2022), which uses a different architecture as the results in Table 1. R-CR uses Wide-ResNet34x10 as backbone, and adopts multiple strong data augmentations. It also adjusts the loss weight along training. For fair comparison, we use the same architecture without multiple augmentation and the curriculum adjust on loss. The results are shown in Table 8, where our method outperforms R-CR on CIFAR-10 and is comparable on CIFAR-100.

Table 8: Comparison with R-CR in partial label learning

Method	CIFAR-10		CIFAR-100	
	0.3	0.5	0.05	0.10
R-CR	97.28±0.02	97.05±0.05	82.77±0.10	82.24±0.07
Ours	97.55±0.07	97.17±0.11	82.46±0.08	82.22±0.05

A recent work on PLL discussed and analyzed the robustness performance of different loss functions, especially the average-based methods (Lv et al., 2023). We perform a similar analysis here for the derived loss function in ILL. Following the notation in Lv et al. (2023), let $\mathbf{s}$ denote the candidate label set, $\mathbf{x}$ as the training instance, g as the probability score from the model, and f as the classifier $f(\boldsymbol{x}) = \arg\max_{i \in \mathcal{Y}} g_i(\boldsymbol{x})$, the average-based PLL can be formulated as:

$$\mathcal{L}_{avg-PLL}(f(\boldsymbol{x}), \mathbf{s}) = \frac{1}{|\mathbf{s}|} \sum_{i \in \mathbf{s}} \ell(f(\boldsymbol{x}), i) \tag{17}$$

Lv et al. (2023) compared different loss functions ℓ on both noise-free and noisy PLL settings, where they find both theoretically and empirically that average-based PLL with *bounded* loss are robust under mild assumptions. Empirical study in Lv et al. (2023) suggests that both *Mean Absolute Error* and *Generalized Cross-Entropy* loss (Zhang & Sabuncu, 2018) that proposed for noisy label learning achieves the best performance and robustness for average-based PLL.

Our solution for PLL can be viewed as an instantiation of the average-based PLL as in (Lv et al., 2023) with:

$$\ell(f(\mathbf{x}), i) = -\bar{g}_i(\mathbf{x}) \log g_i(\mathbf{x}) \tag{18}$$

where $\bar{g}$ is normalized probability over $\mathbf{s}$ with detached gradient. We can further show that the above loss function is bounded for $0 < \ell \leq \frac{1}{e}$ and thus bounded for summation of all classes, which demonstrates robustness, as we show in Table 4.

D.3 Semi-Supervised Learning

D.3.1 Setup

For experiments of SSL, we follow the training and evaluation protocols of USB (Wang et al., 2022d) on image and text classification. To construct the labeled dataset for semi-supervised learning, we uniformly select l/C samples from each class and treat the remaining samples as the unlabeled dataset. For image classification tasks, ImageNet-1K (Deng et al., 2009) Vision Transformers (Dosovitskiy et al., 2020) are used, including CIFAR-100 (Krizhevsky et al., 2009), EuroSAT (Helber et al., 2019), STL-10 (Coates et al., 2011), TissueMNIST (Yang et al., 2021a;b), Semi-Aves (Su & Maji, 2021). For text classification tasks, we adopt BERT (Devlin et al., 2018) as backbone, including IMDB (Maas et al., 2011), Amazon Review (McAuley & Leskovec, 2013), Yelp Review (yel), AG News (Zhang et al., 2015) , Yahoo Answer (Chang et al., 2008). The hyper-parameters strictly follow USB, and are shown in Table 9 and Table 10.

Table 9: Hyper-parameters of **semi-supervised learning** used in vision experiments of USB.

Hyper-parameter	CIFAR-100	STL-10	Euro-SAT	TissueMNIST	Semi-Aves
Image Size	32	96	32	32	224
Model	ViT-S-P4-32	ViT-B-P16-96	ViT-S-P4-32	ViT-T-P4-32	ViT-S-P16-224
Labeled Batch size	16				
Unlabeled Batch size	16				
Learning Rate	5e-4	1e-4	5e-5	5e-5	1e-3
Weight Decay	5e-4				
Layer Decay Rate	0.5	0.95	1.0	0.95	0.65
LR Scheduler	$\eta = \eta_0 \cos(\frac{7\pi k}{16K})$				
Training epochs	20				
Classes	100	10	10	10	200
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	Random Crop, Random Horizontal Flip				
Strong Augmentation	RandAugment (Cubuk et al., 2020)				

Table 10: Hyper-parameters of **semi-supervised learning** NLP experiments in USB.

Hyper-parameter	AG News	Yahoo! Answer	IMDB	Amazom-5	Yelp-5
Max Length	512				
Model	Bert-Base				
Labeled Batch size	4				
Unlabeled Batch size	4				
Learning Rate	5e-5	1e-4	5e-5	1e-5	5e-5
Weight Decay	1e-4				
Layer Decay Rate	0.65	0.65	0.75	0.75	0.75
LR Scheduler	$\eta = \eta_0 \cos(\frac{7\pi k}{16K})$				
Training epochs	10				
Classes	4	10	2	5	5
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	None				
Strong Augmentation	Back-Translation (Xie et al., 2020a)				

D.3.2 Results

In the main paper, we only provide the comparison on CIFAR-100, STL-10, IMDB, and Amazon Review. Here we provide the full comparison in Table 11 and Table 12. From the full results, similar conclusion can be drawn as in the main paper. Our ILL framework demonstrates comparable performance as previous methods.

Table 11: Error rate comparison of different number of labels on CIFAR-100, STL-10, EuroSAT, TissueMNIST, and SemiAves for **semi-supervised learning**. We use USB (Wang et al., 2022d) image classification task results. The best results are indicated in bold. Our results are averaged over 3 independent runs.

Datasets	CIFAR-100		STL-10		EuroSat		TissueMNIST		SemiAves
# Labels	200	400	40	100	20	40	80	400	3959
Pseudo-Label (Lee et al., 2013)	33.99±0.95	25.32±0.29	19.14±1.33	10.77±0.60	25.46±1.36	15.70±2.12	56.92±4.54	50.86±1.79	40.35±0.30
Mean-Teacher (Tarvainen & Valpola, 2017)	35.47±0.40	26.03±0.30	18.67±1.69	24.19±10.15	26.83±1.46	15.85±1.66	62.06±3.43	55.12±2.53	38.55±0.21
VAT (Miyato et al., 2018)	31.49±1.33	21.34±0.50	18.45±1.47	10.69±0.51	26.16±0.96	10.09±0.94	57.49±5.47	51.30±1.73	38.82±0.04
MixMatch (Berthelot et al., 2019b)	38.22±0.71	26.72±0.72	58.77±1.98	36.74±1.24	24.85±4.85	17.28±2.67	55.53±1.51	49.64±2.28	37.25±0.08
ReMixMatch (Berthelot et al., 2019a)	22.21±2.21	16.86±0.57	13.08±3.34	**7.21±0.39**	**5.05±1.05**	5.07±0.56	58.77±4.43	49.82±1.18	**30.20±0.03**
AdaMatch (Berthelot et al., 2021)	22.32±1.73	16.66±0.62	13.64±2.49	7.62±1.90	7.02±0.79	**4.75±1.10**	58.35±4.87	52.40±2.08	31.75±0.13
FixMatch (Sohn et al., 2020)	29.60±0.90	19.56±0.52	16.15±1.89	8.11±0.68	13.44±3.53	5.91±2.02	**55.37±4.50**	51.24±1.56	31.90±0.06
FlexMatch (Zhang et al., 2021b)	26.76±1.12	18.24±0.36	14.40±3.11	8.17±0.78	5.17±0.57	5.58±0.81	58.36±3.80	51.89±3.21	32.48±0.15
Dash (Xu et al., 2021)	30.61±0.98	19.38±0.10	16.22±5.95	7.85±0.74	11.19±0.90	6.96±0.87	56.98±2.93	51.97±1.55	32.38±0.16
CoMatch (Li et al., 2021a)	35.08±0.69	25.35±0.50	15.12±1.88	9.56±1.35	5.75±0.43	4.81±1.05	59.04±4.90	52.92±1.04	38.65±0.18
SimMatch (Zheng et al., 2022)	23.78±1.08	17.06±0.78	11.77±3.20	7.55±1.86	7.66±0.60	5.27±0.89	60.88±4.31	52.93±1.56	33.85±0.08
FreeMatch (Wang et al., 2023)	**21.40±0.30**	**15.65±0.26**	12.73±3.22	8.52±0.53	6.50±0.78	5.78±0.51	58.24±3.08	52.19±1.35	32.85±0.31
SoftMatch (Chen et al., 2023)	22.67±1.32	16.84±0.66	13.55±3.16	7.84±1.72	5.75±0.62	5.90±1.42	57.98±3.66	51.73±2.84	31.80±0.22
Ours	22.06±1.06	17.40±1.04	**11.09±0.71**	8.10±1.02	5.86±1.06	5.74±1.13	57.99±2.16	**50.95±2.03**	33.08±0.26

Table 12: Error rate comparison of different number of labels on IMDB, AG News, Amazon Review, Yahoo Answers, and Yelp Review for **semi-supervised learning**. We use USB (Wang et al., 2022d) text classification task results. Best results are indicated in bold. Our results are averaged over 3 independent runs.

Datasets	IMDB		AG News		Amazon Review		Yahoo Answers		Yelp Review	
# Labels	20	100	40	200	250	1000	500	2000	250	1000
Pseudo-Label (Lee et al., 2013)	45.45±4.43	19.67±1.01	19.49±3.07	14.69±1.88	53.45±1.9	47.00±0.79	37.70±0.65	32.72±0.31	54.51±0.82	47.33±0.20
Mean-Teacher (Tarvainen & Valpola, 2017)	20.06±2.51	13.97±1.49	15.17±1.21	13.93±0.65	52.14±0.52	47.66±0.84	37.09±0.18	33.43±0.28	50.60±0.62	47.21±0.31
VAT (Miyato et al., 2018)	25.93±2.58	11.61±1.79	14.70±1.19	11.71±0.84	49.83±0.46	46.54±0.31	34.87±0.41	31.50±0.35	52.97±1.41	45.30±0.32
MixMatch (Berthelot et al., 2019b)	26.12±6.13	15.47±0.65	13.50±1.51	11.75±0.60	59.54±0.67	61.69±3.32	35.75±0.71	33.62±0.14	53.98±0.59	51.70±0.68
AdaMatch (Berthelot et al., 2021)	8.09±0.99	7.11±0.20	**11.73±0.17**	**11.22±0.95**	46.72±0.72	42.27±0.25	32.75±0.35	30.44±0.31	45.40±0.96	40.16±0.49
FixMatch (Sohn et al., 2020)	7.72±0.33	7.33±0.13	30.17±1.87	11.71±1.95	47.61±0.83	43.05±0.54	**33.03±0.49**	30.51±0.53	46.52±0.94	40.65±0.46
FlexMatch (Zhang et al., 2021b)	7.82±0.77	7.41±0.38	16.38±3.94	12.08±0.73	45.73±1.60	42.25±0.33	35.61±1.08	31.13±0.18	**43.35±0.69**	40.51±0.34
Dash (Xu et al., 2021)	8.34±0.86	7.55±0.35	17.67±3.19	13.76±1.67	47.10±0.74	43.09±0.60	35.26±0.33	31.19±0.29	45.24±2.02	40.14±0.79
CoMatch (Li et al., 2021a)	7.44±0.30	7.72±1.14	11.95±0.76	10.75±0.35	48.76±0.90	43.36±0.21	33.48±0.51	30.25±0.35	45.40±1.12	40.27±0.51
SimMatch (Zheng et al., 2022)	7.93±0.55	**7.08±0.33**	14.26±1.51	12.45±1.37	45.91±0.95	42.21±0.30	33.06±0.20	30.16±0.21	46.12±0.48	40.26±0.62
FreeMatch (Wang et al., 2023)	8.94±0.21	7.95±0.45	12.98±0.58	11.73±0.63	46.41±0.60	42.64±0.06	32.77±0.26	30.32±0.18	47.95±1.45	40.37±1.00
SoftMatch (Chen et al., 2023)	7.76±0.58	7.97±0.72	11.90±0.27	11.72±1.58	45.29±0.95	**42.21±0.20**	33.07±0.31	30.44±0.62	44.09±0.50	39.76±0.13
Ours	**7.32±0.12**	7.64±0.67	14.77±1.59	12.21±0.82	**43.96±0.32**	42.32±0.02	33.80±0.25	30.86±0.17	44.82±0.17	**39.67±0.71**

D.4 Noisy Label Learning

D.4.1 Setup

We conduct experiments of noisy label learning following SOP (Liu et al., 2022). We evaluate the proposed method on both synthetic symmetric/asymmetric noise on CIFAR-10 and CIFAR-100, and more realistic and larger-scale instance noise on Clothing1M and WebVision. To introduce the synthetic symmetric noise to CIFAR-10 and CIFAR-100, we uniformly flip labels for a probability η into other classes. For asymmetric noise, we only randomly flip the labels for particular pairs of classes. For CIFAR-10 and CIFAR-100, we train PreAct-ResNet-18 with SGD using a learning rate of 0.02, a weight decay of $1e-3$, and a momentum of 0.9. We train for 300 epochs with a cosine learning rate schedule and a batch size of 128. For WebVision, we use InceptionResNet-v2 as the backbone and set the batch size to 32. Other settings are similar to CIFAR-10. For Clothing1M, we use ImageNet-1K pre trained ResNet-50 as the backbone. We train it using SGD with an initial learning rate of $2e$-3 for a total of 10 epochs, where the learning rate is reduced by 10 after 5 epochs. In addition, we also conduct experiments on CIFAR-10N and CIFAR-100N. We present the detailed hyper-parameters in Table 13.

D.4.2 Results

In addition to the results regarding noisy label learning provided in the main paper, we also present comparison results on CIFAR-10N and CIFAR-100N (Wei et al., 2021) in Table 14. We include a full comparison on Clothing1M and WebVision, incorporating methods like Co-Teaching, Forward, and CORES, in Table 15. As shown in Table 14, the proposed ILL framework achieves performance comparable to the previous best method, SOP (Liu et al., 2022). On CIFAR-10N, our method yields results very close to SOP in the Random and Aggregate case noise scenarios and surpasses SOP in the

Table 13: Hyper-parameters for **noisy label learning** used in experiments.

Hyper-parameter	CIFAR-10 (CIFAR-10N)	CIFAR-100 (CIFAR-100N)	Clothing1M	WebVision
Image Size	32	32	224	299
Model	PreAct-ResNet-18 (ResNet-34)	PreAct-ResNet-18 (ResNet-34)	ResNet-50 (ImageNet-1K Pretrained)	Inception-ResNet-v2
Batch Size	128	128	64	32
Learning Rate	0.02	0.02	0.002	0.02
Weight Decay	1e-3	1e-3	1e-3	5e-4
LR Scheduler	Cosine	Cosine	MultiStep	MultiStep
Training Epochs	300	300	10	100
Classes	10	100	14	50
Noisy Matrix Scale	1.0	2.0	0.5	2.5

Worst case noise scenario. However, on CIFAR-100N, our method slightly underperforms previous methods, possibly due to the oversimplified noise model utilized in ILL. We believe that a more realistic noise transition model and further tuning of our method could lead to improved performance.

Table 14: Test accuracy comparison of instance independent label noise on CIFAR-10N and CIFAR-100N for **noisy label learning**. The best results are indicated in **bold**, and the second best results are indicated in underline. Our results are averaged over three independent runs with ResNet34 as the backbone.

Dataset	CIFAR-10N						CIFAR-100N	
Noisy Type	Clean	Random 1	Random 2	Random 3	Aggregate	Worst	Clean	Noisy
CE	92.92±0.11	85.02±0.65	86.46±1.79	85.16±0.61	87.77±0.38	77.69±1.55	76.70±0.74	55.50±0.66
Forward (Patrini et al., 2016)	93.02±0.12	86.88±0.50	86.14±0.24	87.04±0.35	88.24±0.22	79.79±0.46	76.18±0.37	57.01±1.03
Co-teaching (Han et al., 2018)	93.35±0.14	90.33±0.13	90.30±0.17	90.15±0.18	91.20±0.13	83.83±0.13	73.46±0.09	60.37±0.27
DivideMix (Li et al., 2020)	-	95.16±0.19	95.23±0.07	95.21±0.14	95.01±0.71	92.56±0.42	-	71.13±0.48
ELR (Liu et al., 2020)	95.39±0.05	94.43±0.41	94.20±0.24	94.34±0.22	94.83±0.10	91.09±1.60	78.57±0.12	66.72±0.07
CORES (Cheng et al., 2020)	94.16±0.11	94.45±0.14	94.88±0.31	94.74±0.03	95.25±0.09	91.66±0.09	73.87±0.16	55.72±0.42
SOP (Liu et al., 2022)	**96.38±0.31**	95.28±0.13	95.31±0.10	95.39±0.11	95.61±0.13	93.24±0.21	**78.91±0.43**	67.81±0.23
Ours	96.21±0.29	**96.06±0.07**	**95.98±0.12**	**96.10±0.05**	**96.40±0.03**	**93.55±0.14**	78.53±0.21	**68.07±0.33**

Table 15: Test accuracy comparison of realistic noisy labels on Clothing1M and WebVision for **noisy label learning**. The best results are indicated in **bold** and the second best results are indicated in underline. Our results are averaged over 3 independent runs. For Clothing1M, we use ImageNet-1K pre trained ResNet50 as the backbone. For WebVision, InceptionResNetv2 is used as the backbone.

Dataset	Clothing1M	WebVision
CE	69.10	-
Forward (Patrini et al., 2016)	69.80	61.10
Co-Teaching (Han et al., 2018)	69.20	63.60
DivideMix (Li et al., 2020)	**74.76**	77.32
ELR (Liu et al., 2020)	72.90	76.20
CORES (Cheng et al., 2020)	73.20	-
SOP (Liu et al., 2022)	73.50	76.60
Ours	74.02±0.12	**79.37±0.09**

D.5 Mixed Imprecise Label Learning

D.5.1 Setup

To create a mixture of various imprecise label configurations, we select CIFAR-10 and CIFAR-100 as base datasets. We first uniformly sample l/C labeled samples from each class to form the labeled dataset and treat the remaining samples as the unlabeled dataset. Based on the labeled dataset, we generate partially labeled datasets by flipping negative labels to false positive labels with the partial ratio q. After obtaining the partial labels, we randomly select η percentage of samples from each class, and recreate the partial labels for them by flipping the ground truth label uniformly to another class. In this setting, unlabeled data, partially labeled data, and noisy labeled data exist simultaneously, which is very challenging and more closely resembles realistic situations. For CIFAR-10, we set $l \in \{1000, 5000, 50000\}$, and for CIFAR-100, we set $l \in \{5000, 10000, 50000\}$. Similarly in the partial label setting, we set $q \in \{0.1, 0.3, 0.5\}$ for CIFAR-10, and $q \in \{0.01, 0.05, 0.1\}$ for CIFAR-100. For noisy labels, we set $\eta \in \{0.1, 0.2, 0.3\}$ for both datasets.

D.5.2 RESULTS

We provide a more complete version of Table 4 in Table 16. On partial noisy labels of CIFAR-10 with partial ratio 0.5 and of CIFAR-100 with partial ratio 0.1, most baseline methods are more robust or even fail to perform. However, our ILL still shows very robust performance with minor performance degradation as increase of noise ratios.

Table 16: Test accuracy comparison of **mixture of different imprecise labels**. We report results of full labels, partial ratio q of $\{0.1, 0.3, 0.5\}$ for CIFAR-10 and $\{0.01, 0.05, 0.1\}$ for CIFAR-100, and noise ratio η of $\{0.1, 0.2, 0.3\}$ for CIFAR-10 and CIFAR-100.

Dataset	# Labels	Partial Ratio q	Noise Ratio η	0	0.1	0.2	0.3
CIFAR-10	50,000	0.1	PiCO+ (Wang et al., 2022b)	95.99±0.03	93.64	93.13	92.18
			IRNet (Lian et al., 2022b)	-	93.44	92.57	92.38
			DALI (Xu et al., 2023)	-	94.15	94.04	93.77
			PiCO+ w/ Mixup (Xu et al., 2023)	-	94.58	94.74	94.43
			DALI w/ Mixup (Xu et al., 2023)	-	95.83	95.86	95.75
			Ours	**96.55±0.08**	**96.47±0.11**	**96.09±0.20**	**95.83±0.05**
		0.3	PiCO+ (Wang et al., 2022b)	95.73±0.10	92.32	92.22	89.95
			IRNet (Lian et al., 2022b)	-	92.81	92.18	91.35
			DALI (Xu et al., 2023)	-	93.44	93.25	92.42
			PiCO+ w/ Mixup (Xu et al., 2023)	-	94.02	94.03	92.94
			DALI w/ Mixup (Xu et al., 2023)	-	95.52	95.41	94.67
			Ours	**96.52±0.12**	**96.2±0.02**	**95.87±0.14**	**95.22±0.06**
		0.5	PiCO+ (Wang et al., 2022b)	95.33±0.06	91.07	89.68	84.08
			IRNet (Lian et al., 2022b)	-	91.51	90.76	86.19
			DALI (Xu et al., 2023)	-	92.67	91.83	89.8
			PiCO+ w/ Mixup (Xu et al., 2023)	-	93.56	92.65	88.21
			DALI w/ Mixup (Xu et al., 2023)	-	95.19	93.89	92.26
			Ours	**96.28±0.13**	**95.82±0.07**	**95.28±0.08**	**94.35±0.08**
CIFAR-100	50,000	0.01	PiCO+ (Wang et al., 2022b)	76.29±0.42	71.42	70.22	66.14
			IRNet (Lian et al., 2022b)	-	71.17	70.10	68.77
			DALI (Xu et al., 2023)	-	72.26	71.98	71.04
			PiCO+ w/ Mixup (Xu et al., 2023)	-	75.04	74.31	71.79
			DALI w/ Mixup (Xu et al., 2023)	-	76.52	76.55	76.09
			Ours	**78.08±0.26**	**77.53±0.24**	**76.96±0.02**	**76.43±0.27**
		0.05	PiCO+ (Wang et al., 2022b)	76.17±0.18	69.40	66.67	62.24
			IRNet (Lian et al., 2022b)	-	70.73	69.33	68.09
			DALI (Xu et al., 2023)	-	72.28	71.35	70.05
			PiCO+ w/ Mixup (Xu et al., 2023)	-	73.06	71.37	67.56
			DALI w/ Mixup (Xu et al., 2023)	-	76.87	75.23	74.49
			Ours	**76.95±0.46**	**77.07±0.16**	**76.34±0.08**	**75.13±0.63**
		0.1	PiCO+ (Wang et al., 2022b)	75.55±0.21	-	-	-
			IRNet (Lian et al., 2022b)	-	-	-	-
			DALI (Xu et al., 2023)	-	-	-	-
			PiCO+ w/ Mixup (Xu et al., 2023)	-	-	-	-
			DALI w/ Mixup (Xu et al., 2023)	-	-	-	-
			Ours	**76.41±1.02**	**75.50±0.54**	**74.67±0.30**	**73.88±0.60**