Estimating Noisy Class Posterior with Part-level Labels for Noisy Label Learning

Supplementary Material

A. Classifier training with PLM

In Eq. (4) of main paper, we discussed the empirical risk for estimating the noisy class posterior and the single-to-multiple transition matrix. In this supplementary material, we will provide a detailed discussion on how to train a consistent classifier using PLM and loss correction techniques [6].

As discussed in the main paper, the classification task aims to learn a classifier $f : \mathcal{X} \rightarrow \mathcal{C}$ that maps each instance $\boldsymbol{x}_i$ to its corresponding label y_i. Given the network for estimating the clean class posterior as $g : \mathcal{X} \rightarrow \mathcal{R}^c$, the classifier can be represented as follows: $f(\boldsymbol{x}) = \arg\max_{i \in \mathcal{C}} g_i(\boldsymbol{x})$. Here, $g_i(\boldsymbol{x})$ refers to the i-th element of the vector $g(\boldsymbol{x})$, which represents the estimated probability $\hat{P}(Y = i|X = \boldsymbol{x})$. Given a noisy dataset $\tilde{\mathcal{D}} = \{(\boldsymbol{x}_i, \tilde{y}_i)\}_{i=1}^n$, the empirical risk of the classifier is defined as:

$$\tilde{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell_1(f(\boldsymbol{x}_i), \tilde{y}_i), \tag{1}$$

where ℓ_1 denotes a classification loss. Loss correction methods typically introduce a transition matrix $T(\boldsymbol{x})$ to establish a connection between the posterior of the noisy and clean classes. This allows training a clean classifier by minimizing the empirical risk with noisy dataset. Based on existing loss correction methods, the noise transition matrix $T(\boldsymbol{x})$ can be estimated, and we have $P(\tilde{\boldsymbol{Y}}|X = \boldsymbol{x}) = T(\boldsymbol{x})^\top P(\boldsymbol{Y}|X = \boldsymbol{x})$. Let the noisy class posterior estimation network be denoted as $g^e : \mathcal{X} \rightarrow \mathcal{R}^c$ where $g_i^e(\boldsymbol{x}) = P(\tilde{Y} = i|X = \boldsymbol{x})$. The noisy class classifier $f^e(\boldsymbol{x})$ can be represented as:

$$f^e(\boldsymbol{x}) = \arg\max_{i \in \mathcal{C}} g_i^e(\boldsymbol{x}) = \arg\max_{i \in \mathcal{C}} (T(\boldsymbol{x})^\top g)_i(\boldsymbol{x}). \tag{2}$$

Therefore, the empirical risk in loss correction methods can be expressed as:

$$\tilde{R}(g) = \frac{1}{n} \sum_{i=1}^{n} \ell_1(f^e(\boldsymbol{x}_i), \tilde{y}_i). \tag{3}$$

By minimizing this loss it is possible to construct classifier-consistent algorithms.

We denote the part-level labels estimation network as $g^p : \mathcal{X} \rightarrow \mathcal{R}^c$ where $g_i^p(\boldsymbol{x}) = P(Y_i' = 1|X = \boldsymbol{x})$. Given the single-to-multiple transition matrix $U(\boldsymbol{x})$ where $U_{ij}(\boldsymbol{x}) = P(Y_j' = 1|\tilde{Y} = i, X = \boldsymbol{x})$, the part-level multi-label classifier $f^p(\boldsymbol{x})$ can be represented as:

$$\begin{aligned} f^p(\boldsymbol{x}) &= \{i|g_i^p(\boldsymbol{x}) > \frac{1}{2}\} = \{i|(U(\boldsymbol{x})^\top g^e)_i(\boldsymbol{x}) > \frac{1}{2}\} \\ &= \{i|(U(\boldsymbol{x})^\top T(\boldsymbol{x})^\top g)_i(\boldsymbol{x}) > \frac{1}{2}\}. \end{aligned} \tag{4}$$

Similarly, given a dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$ with multiple part-level labels, the empirical risk of the training with part-level labels is defined as:

$$R'(f^p) = \frac{1}{n} \sum_{i=1}^{n} \ell_2(f^p(\boldsymbol{x}_i), \boldsymbol{y}_i'). \tag{5}$$

where ℓ_2 denotes a multi-label classification loss.

Then, the empirical risk of the joint training framework

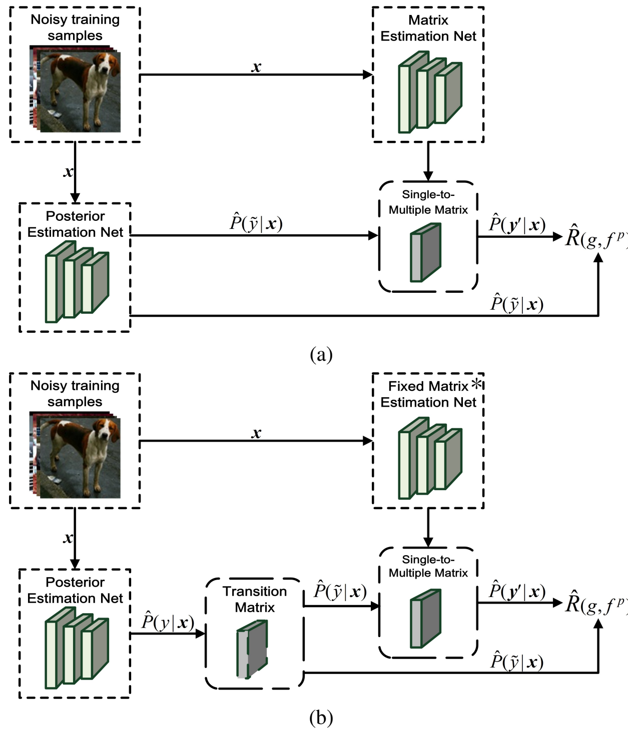

Figure 1. Illustration of neural network training using PLM. (a) Utilizing PLM for estimating the noisy class posterior while simultaneously training the matrix estimation network. (b) Fixing the matrix estimation network (with ”*”) and integrating loss correction method to facilitate noisy label learning.

Algorithm 1 PLM training framework

Input: Noisy training dataset $\mathcal{D}$, noise transition matrix $T(\boldsymbol{x})$ derived from existing methods.
Output: Classifier model f.

Minimize a classification loss to learn a labeling classifier f^l from $\mathcal{D}$.

Obtain the set of sub-instances $\mathcal{S}$ by cropping the instances in $\mathcal{D}$.

Construct multi-labels by using f^l to label the sub-instances in $\mathcal{S}$.

Train a single-to-multiple transition matrix estimation network g^u by minimizing the loss defined in Eq. (4) of the main paper.

Fix the parameters of g^u, set $g^e(\boldsymbol{x}) = (T(\boldsymbol{x})^\top g)(\boldsymbol{x})$, then minimize the Eq. (4) of the main paper with updated g^u to optimize g.

Obtain the classifier $f(\boldsymbol{x}) = \arg\max_{i \in \mathcal{C}} g_i(\boldsymbol{x})$. Here, $g_i(\boldsymbol{x})$ represents the i-th element of the network output vector $g(\boldsymbol{x})$.

return Optimized classifier f.

is defined as:

$$\begin{aligned} \hat{R}(g, f^p) &= \frac{1}{2}(\tilde{R}(g) + R'(f^p)) \\ &= \frac{1}{2n}\sum_{i=1}^{n}[\ell_1(f^e(\boldsymbol{x}_i), \tilde{y}_i) + \ell_2(f^p(\boldsymbol{x}_i), \boldsymbol{y}'_i)]. \end{aligned} \tag{6}$$

We minimize the empirical risk to obtain a robust classifier. The training process for the single-to-multiple transition matrix is depicted in Figure 1a, in which we achieve it by minimizing the empirical risk defined in Eq. (4) of the main paper. The training of the classifier is illustrated in Figure 1b, where we keep the trained matrix estimation network fixed and combine it with the noise transition matrix obtained through existing loss correction methods. We then optimize the empirical risk discussed before for training. The training procedure is outlined concisely in Algorithm 1.

In Section 4.3 of the main paper, we have conducted a comparison of performance using various matrix estimation methods. More specifically, for PLM-F, PLM-D, and PLM-V, we adopted the matrix estimation techniques outlined in Forward [6], Dual-T [10], and VolMinNet [5], respectively. Subsequently, we minimize the empirical risk as defined in Eq. (6). For PLM-R, we combined PLM with T-Revision [8] and introduced the slack variable ΔT, then $f^e(\boldsymbol{x})$ in Eq. (2) can be modified to

$$f^{er}(\boldsymbol{x}) = \arg\max_{i \in \mathcal{C}}((T(\boldsymbol{x}) + \Delta T)^\top g)_i(\boldsymbol{x}). \tag{7}$$

Following T-Revision, we also incorporated a importance reweighting strategy. The minimized empirical risk of PLM-R is defined as follows:

$$\hat{R}(f^{er}, f^p) = \frac{1}{2n}\sum_{i=1}^{n}[w\ell_1(f^{er}(\boldsymbol{x}_i), \tilde{y}_i) + \ell_2(f^p(\boldsymbol{x}_i), \boldsymbol{y}'_i)], \tag{8}$$

where $w = \frac{g_{y_i}(\boldsymbol{x}_i)}{((T(\boldsymbol{x})+\Delta T)^\top g)_{y_i}(\boldsymbol{x}_i)}$ denotes the weight.

B. Identifiability of single-to-multiple transition matrix

In the main paper, we introduce a brand-new single-to-multiple transition matrix. In this section, we will discuss the identifiability of this transition matrix. Specifically, regarding $P(\boldsymbol{Y}'|\boldsymbol{x}) = U^\top(\boldsymbol{x})P(\tilde{\boldsymbol{Y}}|\boldsymbol{x})$, when the matrix $U(\boldsymbol{x})$ is unconstrained, the following issue arises: there exists an infinite number of non-singular matrices $Q \in \mathbb{R}^{c \times c}$ such that

$$P(\boldsymbol{Y}'|\boldsymbol{x}) = (U^\top(\boldsymbol{x})Q)(Q^{-1}P(\tilde{\boldsymbol{Y}}|\boldsymbol{x})). \tag{9}$$

This situation emerges from the network training process in joint training framework of Section 3.4:

$$g^p(\boldsymbol{x}) = g^u(\boldsymbol{x})g^e(\boldsymbol{x}), \tag{10}$$

where $g^p(\boldsymbol{x})$, $g^e(\boldsymbol{x})$ and $g^u(\boldsymbol{x})$ correspond to the estimates of part-level labels, noisy class posterior, and the matrix respectively. The specific concern appears to center on the scenario where $g^p(\boldsymbol{x}) = \hat{P}(\boldsymbol{Y}'|\boldsymbol{x})$ and $g^e(\boldsymbol{x}) = Q^{-1}\hat{P}(\tilde{\boldsymbol{Y}}|\boldsymbol{x})$, yielding $g^u(\boldsymbol{x}) = U^\top(\boldsymbol{x})Q \neq U^\top(\boldsymbol{x})$.

To address this issue, we employ joint training to simultaneously utilize noisy labels and part-level labels for optimizing both g^e and g^p. More precisely, g^e is directly guided by $\tilde{Y}$, aligning with a coarse $g^e(\boldsymbol{x}) = \hat{P}(\tilde{\boldsymbol{Y}}|\boldsymbol{x})$, while $g^p(\boldsymbol{x})$ is supervised by Y' to meet $g^p(\boldsymbol{x}) = \hat{P}(\boldsymbol{Y}'|\boldsymbol{x})$. This dual supervision constrains $g^u(\boldsymbol{x})$ to comply with $\hat{P}(\boldsymbol{Y}'|\boldsymbol{x}) = g^u(\boldsymbol{x})\hat{P}(\tilde{\boldsymbol{Y}}|\boldsymbol{x})$, resulting in $g^u(\boldsymbol{x}) = \hat{U}(\boldsymbol{x})^\top$. This means that the potential scenario, where Q leads to $g^e(\boldsymbol{x}) = Q^{-1}\hat{P}(\tilde{\boldsymbol{Y}}|\boldsymbol{x})$ and then $g^u(\boldsymbol{x}) = \hat{U}(\boldsymbol{x})^\top Q$, is preemptively negated through supervision from $\tilde{Y}$. Therefore, during training with the joint framework, the matrix's identifiability is ensured. This approach also echoes the matrix estimation strategy presented in MEIDTM [1].

C. Analysis of time complexity

In the main paper, we introduced additional modules to aid in estimating the noisy class posterior, which to some extent increases the algorithm's time complexity. Therefore, in this section, we will discuss the time complexity and efficiency of the proposed method in comparison to our baseline model (Forward [6]).

Let us assume that the time complexity for training the baseline model for one epoch is denoted as $O(T)$, and the time complexity for making predictions on the entire training set is $O(P)$. Additionally, the introduction of an extra noise transition matrix layer contributes an additional time complexity of $O(E)$. Considering Forward as the baseline, the time complexity of the proposed method can be expressed as $O(e_1T+cP+e_2(2T+E)+e_3(T+P+2E))$, where e_1, e_2, and e_3 represent the number of epochs for the annotator, transition matrix estimator, and classifier training, respectively, and c indicates the cropping frequency.

For comparison, the time complexity of the Forward method is expressed as $O(e_4T+P+e_5(T+E))$, where e_4 and e_5 represent the number of epochs for the anchor estimation network and classifier training, respectively. For the sake of facilitating comparison, we assume that each section underwent an equal number of training epochs e, i.e., $e = e1 = e2 = e3 = e4 = e5$. The complexity of the proposed method is represented as $O(4eT + (c+e)P + (1+2e)E)$, whereas the complexity of the Forward method is denoted as $O(2eT + P + eE)$. Then, the transition matrix can be considered as a noise adaptation layer with fixed parameters, which has a total of c^2 parameters where c represents the number of categories. In comparison, deep neural networks have a much larger number of parameters. Taking Resnet-18 as an example, it has a total of 11.7M parameters. Therefore, in this paper and in most cases, we have $T << P$ and $T << E$. Consequently, we can simplify the two computational complexities to $O(4eT + (c+e)P)$ and $O(2eT + P)$. Since the training process involves backpropagation and gradient computation, it takes more time than the prediction process, leading to $T > P$. Additionally, in this paper, the number of pruning iterations satisfies $e \gg c$. As a result, the computational complexity of the proposed method follows $O(4eT + (c+e)P) < O(6eT)$, and the Forward follows $O(2eT+P) > O(2eT)$. Hence, under the assumption of setting the same number of epochs in each stage, the time overhead of the proposed method should be less than three times that of the Forward. Additionally, for the purpose of evaluating the efficiency of our approach, we conducted a comparison of the code's runtime based on the CIFAR-10 dataset.

Table 1. The time consumption of PLM and Forward (used as the baseline).

Method	Time Consumption (min)
PLM	35.56
Forward	19.21

Analysis and experiments indicate that our approach significantly enhances the performance of noisy label learning (NLL), with only a linear increase in time consumption.

D. Analysis of cropping strategies

The instance cropping method is related to the multi-labeling of the proposed approach. In the paper, we selected the four corners and the central part of the image data for cropping and determined the cropping size through empirical analysis on the validation set. Table 2 displays the experimental results of different cropping sizes on CIFAR-10 data with sym-50% noise, and we additionally attempted two other cropping strategies. The cropping strategies used in table 2 are as follows: the uniform strategy involves five uniform crops at the four corners and center of the image as used in the paper. The random strategy entails five crops at random positions. The emphasized strategy constructs two sub-instances based on feature emphasis, with one sub-instance masking the top emphasized number of features and the other sub-instance masking the remaining features.

Table 2. The classification accuracy (expressed in percentage) with different cropping sizes and strategies.

Size	Uniform	Random	Emphasized
9	83.58 ± 0.45	83.42 ± 0.86	84.14 ± 0.59
36	83.80 ± 0.31	83.40 ± 0.55	84.24 ± 0.29
81	82.69 ± 2.32	83.81 ± 0.54	84.48 ± 0.47
144	83.17 ± 0.91	83.81 ± 0.39	84.19 ± 0.42
256	83.49 ± 0.95	83.40 ± 0.34	84.28 ± 0.86
361	83.62 ± 0.31	83.65 ± 0.74	84.32 ± 0.46
484	84.99 ± 0.40	84.36 ± 0.32	84.24 ± 0.61
625	85.08 ± 0.16	83.97 ± 0.72	84.26 ± 0.57
784	84.24 ± 0.24	83.91 ± 0.70	84.47 ± 0.33

The emphasized strategy demonstrates superior performance and displays enhanced stability, suggesting the potential for further refinement of cropping strategies in the context of NLL classification, as discussed in Section 5 of the paper. Furthermore, within the established cropping strategy, the method shows robustness to the cropping size.

E. Visualization of focused features

In Figure 2, we employ a visualization approach to provide a visual interpretation of the effectiveness of the PLM method. The STL-10 [2] dataset is used for visualization purposes. In Figure 2b, it is shown that when the labels contain noise, the network emphasizes the background region associated with those labels. Consequently, the model tends to overfit to the noise, hindering the network's ability to learn features that truly capture the distinctive characteristics of the instances. As a result, the estimation of the posterior for the noisy labels becomes excessively confident. As depicted in Figure 2c, removing the overemphasized features through cropping effectively redirects the model's attention to other more informative features. By generating

CVPR
#1943

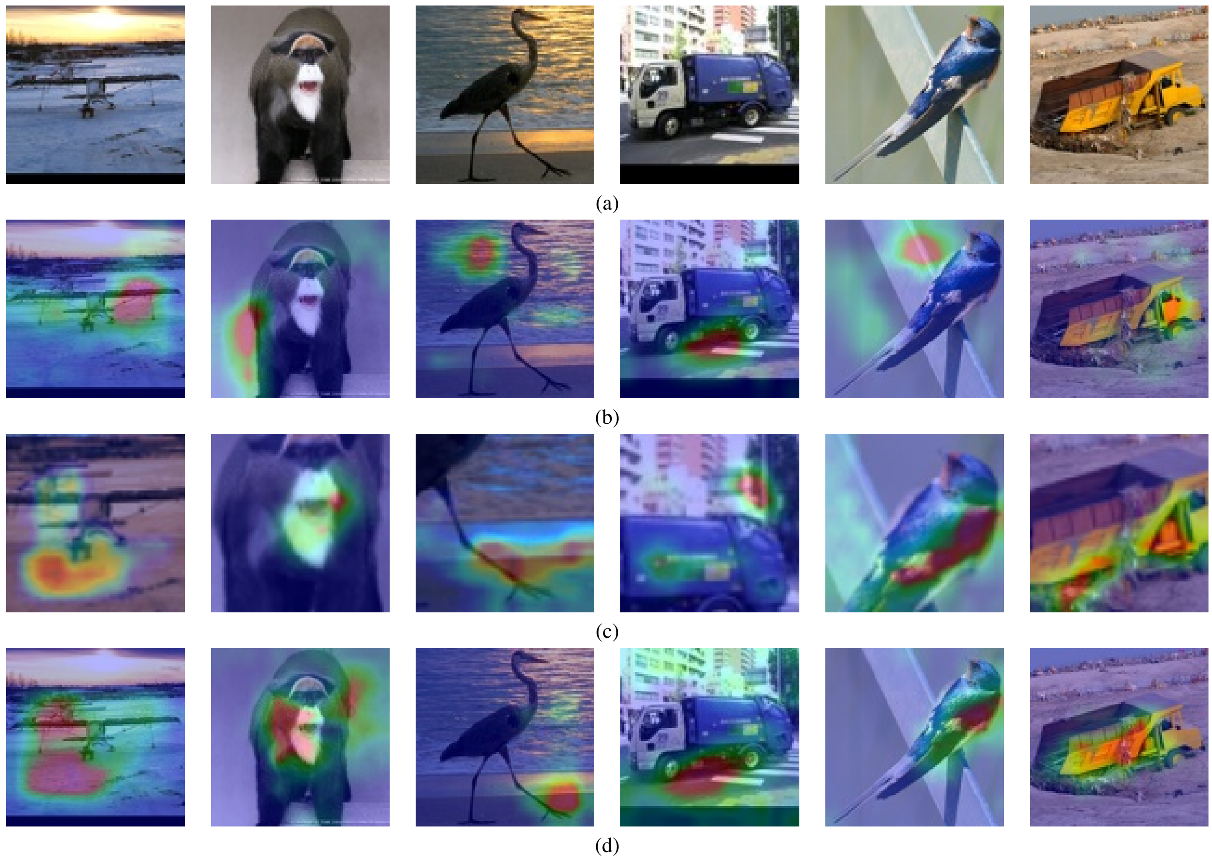

Figure 2. Illustration of class activation maps (CAM) for overemphasized region correction: the highlighted area (with more intense red color) indicates the emphasized area of a model trained from noisy labels. (a) Original images with noisy labels: car, bird, dog, monkey, ship, deer. (b) CAMs for estimating noisy class posterior by the classifier. (c) CAMs when excluding the overemphasized regions after cropping. (d) CAMs for estimating noisy class posterior by the model after PLM training.

labels associated with these features and providing additional supervised information during network training, the network can focus on more diverse features. As shown in Figure 2d, compared to Figure 2b, the network pays more attention to object-relevant features.

F. Combination with state-of-the-art methods

In this section, we conbine and compare our proposed framework with different state-of-the-art (SOTA) methods to further validate the effectiveness of our approach. In the Section 4 of the main paper, we mentioned that we did not compare PLM with SOTA methods. This is because these methods incorporate a lot of robust learning strategies to achieve better empirical performance, while our method focuses solely on enhancing the noisy class posterior estimation to assist in building a classifier-consistent algorithm. To explore whether our method can flexibly combine with these robust learning strategies to achieve improved classification performance, we conbine and compare our proposed method with SOTA methods using different strategies, as detailed in Table 3 and 4. Specifically, in Table 3, we compare our method with following two representative SOTA methods on the CIFAR-10 and CIFAR-100 datasets: (1) DivideMix [4], a method that combines data augmentation, label selection, co-training, semi-supervised learning, and pseudo-labeling strategies; (2) CTRR [12], a method based on contrastive learning strategies that constructs a regularization function. Besides, in Table 4, we compare PLM with following two methods designed for instance-dependent noise on the CIFAR-10 dataset: (1) BLTM [9], a method using deep neural networks and bayes optimal labels to estimate transition matrix; (2) CausalNL [11], a method based on a structural causal framework. We combine PLM with these methods to verify its complementary effects. In the combination with DivideMix (named PLM_DivideMix), we introduce an additional loss term based on Eq. (4) of the main paper to the selected labeled samples. In the combination with CTRR and CausalNL (named PLM_CTRR and PLM_CausalNL), we modify the cross-entropy loss term of their loss function to Eq. (4) of the main paper. In the com-

Table 3. The average classification accuracy and standard deviation (expressed in percentage) across five trials under various synthetic noisy label settings. The better classification accuracy is indicated in **bold**.

	CIFAR-10				CIFAR-100			
	Sym-20%	Sym-50%	Pair-20%	Pair-45%	Sym-20%	Sym-50%	Pair-20%	Pair-45%
CTRR	93.02 $\pm$ 0.12	84.96 $\pm$ 1.12	92.81 $\pm$ 0.27	68.54 $\pm$ 1.81	71.61 $\pm$ 0.64	65.53 $\pm$ 0.46	69.94 $\pm$ 0.36	46.17 $\pm$ 0.70
PLM_CTRR	**93.16 $\pm$ 0.36**	**86.59 $\pm$ 0.26**	**92.96 $\pm$ 0.36**	**78.56 $\pm$ 1.71**	**72.06 $\pm$ 0.26**	**66.00 $\pm$ 0.56**	**70.44 $\pm$ 0.33**	**46.84 $\pm$ 0.52**
DivideMix	95.71 $\pm$ 0.47	94.77 $\pm$ 0.06	92.65 $\pm$ 0.38	68.67 $\pm$ 1.95	76.72 $\pm$ 0.31	73.12 $\pm$ 0.30	76.62 $\pm$ 0.25	47.01 $\pm$ 1.02
PLM_DivideMix	**96.06 $\pm$ 0.19**	**95.12 $\pm$ 0.18**	**96.01 $\pm$ 0.07**	**76.27 $\pm$ 1.13**	**79.96 $\pm$ 0.14**	**74.20 $\pm$ 0.40**	**79.93 $\pm$ 0.18**	**47.19 $\pm$ 0.88**

Table 4. The average classification accuracy and standard deviation (expressed in percentage) across five trials on the CIFAR-10 dataset with instance-dependent noise settings. The better classification accuracy is indicated in **bold**.

	IDN-20%	IDN-30%	IDN-40%	IDN-50%
BLTM	76.70 $\pm$ 0.55	72.12 $\pm$ 0.59	65.44 $\pm$ 1.01	56.77 $\pm$ 0.75
PLM_BLTM	**89.73$\pm$ 0.22**	**87.47 $\pm$ 0.50**	**84.40 $\pm$ 0.84**	**76.28 $\pm$ 3.80**
CausalNL	79.66 $\pm$ 0.38	76.58 $\pm$ 0.46	72.86 $\pm$ 0.43	67.75 $\pm$ 1.15
PLM_CausalNL	**81.44 $\pm$ 0.38**	**78.86 $\pm$ 0.92**	**75.52 $\pm$ 0.38**	**73.20 $\pm$ 1.06**

bination with BLTM (named PLM_BLTM), we use the transition matrix estimated in BLTM. The remaining settings are consistent with those in Section 4 of the main paper.

The experimental results demonstrate that our approach can integrate with SOTA methods which rely on complex robust learning strategies, improving their classification performance. The improvement occurs even though these methods do not explicitly require the explicit estimation of noisy class posteriors. This could be because, during the process of estimating noisy class posteriors using PLM, the richer supervised information assists the model in learning more reasonable representations that reflect the instance characteristic.

G. Analysis of transition matrix estimation

In this section, we aim to validate the assistance of PLM in transition matrix estimation error. In the experiments presented in Table 5, we modified the strategy of noisy class posterior estimation used in the most basic transition matrix estimation method Forward [6] to PLM's strategy. The results indicate that PLM can help in transition matrix estimation by reducing the error in estimating noisy class posterior.

Table 5. The average errors and standard deviation of transition matrix estimation across five trials on the CIFAR-10 dataset. The lower error is indicated in **bold**.

	Sym-20%	Sym-50%	Pair-20%	Pair-45%
Forward	0.35 $\pm$ 0.01	0.60 $\pm$ 0.12	0.27 $\pm$ 0.01	0.74 $\pm$ 0.03
PLM_Forward	**0.16 $\pm$ 0.04**	**0.32 $\pm$ 0.06**	**0.23 $\pm$ 0.01**	**0.62 $\pm$ 0.04**

H. Experiments on real-world dataset

In the main paper, we compared the experimental results on the real-world dataset Clothing1M. To further illustrate the performance of PLM on real-world datasets, we compared the results on the Animal-10N [7] dataset in the Table 6. Animal-10N consists of 50,000 noisy samples for training and 5,000 clean samples for testing. We selected 10% of the training set as the validation set. We use the SGD optimizer and cosine learning rate decay strategy to train the network, with an initial learning rate 10^{-2}, weight decay of 10^{-2}, and momentum of 0.9. The backbone and other settings are the same as InstanceGM [3]. The results further demonstrate that PLM can more effectively handle real-world noise.

Table 6. Accuracy on the Animal-10N benchmark. The baseline results and experimental settings refer to InstanceGM. The better classification accuracy is indicated in **bold**.

Method	CE	Dropout	SELFIE	PLC	Nested	InstanceGM	PLM
Acc. (%)	79.4	81.3	81.8	83.4	81.3	84.6	**85.08**

References

[1] De Cheng, Tongliang Liu, Yixiong Ning, Nannan Wang, Bo Han, Gang Niu, Xinbo Gao, and Masashi Sugiyama. Instance-dependent label-noise learning with manifold-regularized transition matrix estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16630–16639, 2022. 2

[2] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. 3

[3] Arpit Garg, Cuong Nguyen, Rafael Felix, Thanh-Toan Do, and Gustavo Carneiro. Instance-dependent noisy label learning via graphical modelling. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2288–2298, 2023. 5

[4] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, pages 1–13, 2019. 4

[5] Xuefeng Li, Tongliang Liu, Bo Han, Gang Niu, and Masashi Sugiyama. Provably end-to-end label-noise learning without anchor points. In *International Conference on Machine Learning*, pages 6403–6413. PMLR, 2021. 2

[6] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952, 2017. 1, 2, 5

[7] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. SELFIE: Refurbishing unclean samples for robust deep learning. In *ICML*, 2019. 5

[8] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, 32:1–12, 2019. 2

[9] Shuo Yang, Erkun Yang, Bo Han, Yang Liu, Min Xu, Gang Niu, and Tongliang Liu. Estimating instance-dependent bayes-label transition matrix using a deep neural network. In *International Conference on Machine Learning*, pages 25302–25312. PMLR, 2022. 4

[10] Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. Dual t: Reducing estimation error for transition matrix in label-noise learning. *Advances in neural information processing systems*, 33: 7260–7271, 2020. 2

[11] Yu Yao, Tongliang Liu, Mingming Gong, Bo Han, Gang Niu, and Kun Zhang. Instance-dependent label-noise learning under a structural causal model. *Advances in Neural Information Processing Systems*, 34:4409–4420, 2021. 4

[12] Li Yi, Sheng Liu, Qi She, A Ian McLeod, and Boyu Wang. On learning contrastive representations for learning with noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16682–16691, 2022. 4