# PLReMix: Combating Noisy Labels with Pseudo-Label Relaxed Contrastive Representation Learning (Supplementary Material)

Xiaoyu Liu, Beitong Zhou, Zuogong Yue, Cheng Cheng*
Huazhong University of Science and Technology
{lxysl, zhoubt, z_yue, c_cheng}@hust.edu.cn

## A. Pseudocode of Pseudo-Label Relaxed loss

Algorithm 1 provides the pseudocode of our proposed PLR loss. For a mini-batch of samples $\boldsymbol{x}$, we first calculate their model prediction probabilities using the first network. Based on the predicted probabilities, we then determine if there are overlapping top$_\kappa$ predictions denoted as *conflicts* and use it to identify feasible negative samples. The reliable negative set $\mathcal{N}_i$ for each sample is built in the form of a contrastive mask, indicating which pairs are negative, positive, or neglected when conducting CRL. Finally, we transform the samples $\boldsymbol{x}$ into two strong augmented views, which are then utilized to train a contrastive loss (using the mask to identify the positive and reliable negative pairs) on the other network.

## B. Training Details

**CIFAR-10/100** We use the PreAct ResNet-18 [3] network on CIFAR 10/100 and train it using an SGD optimizer with a weight decay of 5e-4, a momentum of 0.9, and a batch size of 64. We choose $\lambda_{\mathcal{U}}$ from $\{0, 25, 50, 150\}$ following the previous work [4], although experiments show that our method is not sensitive to this parameter. We set the initial learning rate to 0.02 and reduce it by a factor of 10 after 200 epochs. The warmup period is 10 for CIFAR-10 and 15 for CIFAR-100. We use the Flat version of the proposed PLR loss on these two datasets. To avoid conflict early and gain robust representation later, we reduce $\kappa$ from 3 to 2 to 1 at the epochs of 40 and 70. We empirically set $\tau_1 = 0.5$ for $\mathcal{X}$, especially when the label noise is asymmetric; otherwise, set $\tau_1 = 1/C$.

**Tiny-ImageNet** We train a PreAct ResNet 18 [3] network with an SGD optimizer and a batch size of 128. We set the initial learning rate to 0.01 and reduce it by a factor of 10 after 200 epochs. $\lambda_{\mathcal{U}}$ is chosen from $\{0, 25, 50, 150\}$ and $\beta$ is 0.5. We use the Flat version of the proposed PLR loss on this dataset, and $\kappa$ is reduced at the epochs of 40 and 70.

**Clothing1M** We train a ResNet50 network with weights pre-trained on ImageNet [2] for 80 epochs with a learning rate of 5e-3, a weight decay of 1e-3, and a batch size of 64. We use $\lambda_{\mathcal{U}} = 0$ and $\beta = 0.5$ on the Clothing1M dataset. The network is trained for 100 epochs with a warmup period of 1 epoch. We reduce $\kappa$ at the epochs of 15 and 30. We fix the backbone parameters and use strong augmented samples to warmup the classification and projection heads. The learning rate is reduced by a factor of 10 after 40 epochs.

We list hyperparameters used in our method for various datasets in Tab. 1 and Tab. 2. To maintain simplicity, we keep most of the hyperparameters consistent across all datasets. Additionally, to facilitate comparison with DivideMix [4], we also maintain consistency with most of the hyperparameters used in their approach. It is noteworthy that our PLR loss is insensitive to the hyperparameter $\kappa$, as is further discussed in the paper. We didn't carefully tune it and empirically set it to dynamically decrease for full utilization of negative pairs.

**WebVision** We train an InceptionResnet V2 [6] from scratch with a learning rate of 0.015, a weight decay of 5e-4, a batch size of 96, and a warmup of 2 epochs. We set $\lambda_{\mathcal{U}} = 0$ and $\beta = 0.5$, and reduce $\kappa$ at the epochs of 15 and 30.

**Multi-crop Strategy** Multi-crop strategy [1] is an efficient augmentation method used in CRL. Images are cropped into multiple smaller views of varying sizes, which can be viewed as positive pairs in CRL. This strategy increases the diversity of positive pairs without incurring significant additional computational costs. In the WebVision dataset, we resize the images to a size of 320, then randomly crop and resize them into two large views with a size of 224 and six small views with a size of 128.

**Algorithm 1** Pseudocode of computing PLR loss in a PyTorch-like style.

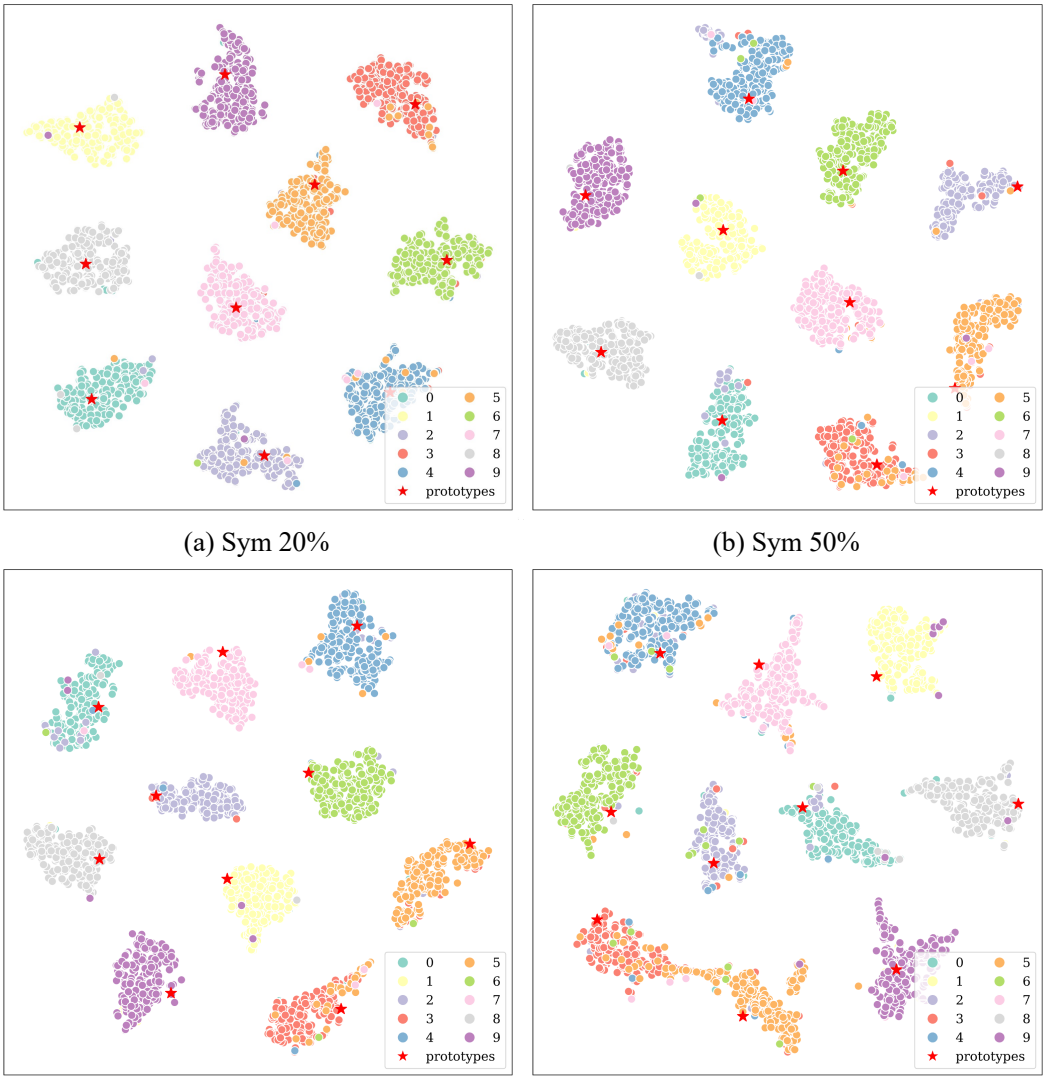
```
# net0, net1: two identical networks with backbone f, classification head g, and projection head h
# PLRLoss: contrastive loss, Eqn.(9), which takes a mask as parameter
# k: hyperparameter kappa in Eqn.(6)

def build_mask(x, y, net):
   z = net.g.forward(net.f.forward(x)) # model outputs
   indices_k = torch.topk(z, k, dim=1)[1] # top k indices of model outputs

   # tops: assign 1 to the top k indices, 0 to the rest
   tops = torch.zeros(len(x), C)
   tops = torch.scatter(tops, 1, indices_k, 1)
   tops = torch.scatter(tops, 1, y.unsqueeze(1), 1) # append labels to top k
   # intersection, Eqn.(8), where 'conflicts' equals 0 are feasible negative pairs
   conflicts = torch.matmul(tops, tops.t())

   # contrastive mask, where negative pairs are -1, positive pairs are 1, neglect pairs are 0
   mask = torch.where(conflicts == 0, -1, 0)
   mask = torch.where(eye(len(x)) == 1, 1, mask)
   return mask


for (x, y) in loader: # load a minibatch of samples and labels
   mask = build_mask(x, y, net0) # get mask from one network

   x1 = aug(x) # random strong augmentation
   x2 = aug(x) # another strong augmentation

   # train the other network
   f1 = net1.h.forward(net1.f.forward(x1))
   f2 = net1.h.forward(net1.f.forward(x2))

   f = torch.cat([f1.unsqueeze(1), f2.unsqueeze(1)], dim=1)
   plr_loss = InfoNCELoss()(f, mask=mask)

   # if other losses exist, sum all losses up, then backward and update
   plr_loss.backward()
   update(net1.params) # train the other network
```

Table 1. Hyperparameter settings of our proposed method on different datasets.

| **Hyperparameters** | **CIFAR-10** | **CIFAR-100** | **Tiny-ImageNet** | **Clothing1M** | **WebVision** |
|---|---|---|---|---|---|
| **Initial Learning Rate** | 0.02 | 0.02 | 0.01 | 0.004 | 0.015 |
| **Momentum** | 0.9 | | | | |
| **Weight Decay** | 0.0005 | 0.0005 | 0.0005 | 0.001 | 0.0005 |
| **Batch Size** | 128 | 128 | 256 | 64 | 96 |
| **Epochs** | 400 | 400 | 400 | 80 | 150 |
| **warmup epochs** | 10 | 15 | 10 | 1 | 2 |
| $\boldsymbol{\beta}$ | 4 | 4 | 0.5 | 0.5 | 0.5 |
| $\boldsymbol{\lambda_i}$ | 1 | | | | |
| $\boldsymbol{T}$ | 0.5 | | | | |
| $\boldsymbol{\alpha}$ | 0.5 | | | | |
| $\boldsymbol{\tau}$ | 1 | | | | |
| $\boldsymbol{\tau_s}$ | 0.1 | | | | |

Table 2. The value of hyperparameter $\lambda_{\mathcal{U}}$ on different datasets, following previous work [4].

| Hyperparameter | Dataset | Noise Ratio $r$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | Sym | | | | | Asym |
| | | **0** | **20%** | **50%** | **80%** | **90%** | **40% / 45%** |
| $\lambda_{\mathcal{U}}$ | **CIFAR-10** | - | 0 | 25 | 25 | 50 | 0 |
| | **CIFAR-100** | - | 25 | 150 | 150 | 150 | 0 |
| | **Tiny-ImageNet** | 0 | 30 | 200 | 300 | - | 0 |
| | **Clothing1M** | 0 | | | | | |
| | **WebVision** | 0 | | | | | |




(a) Sym 20% (b) Sym 50%

(c) Sym 80% (d) Sym 90%

Figure 1. T-SNE visualizations of features and prototypes. The subplots show class distributions after training networks for 400 epochs on the CIFAR-10 dataset with various noise ratios: (a) 20% symmetric, (b) 50% symmetric, (c) 80% symmetric, (d) 90% symmetric. Our proposed method effectively learns robust representations and class prototypes even with high noise ratios.

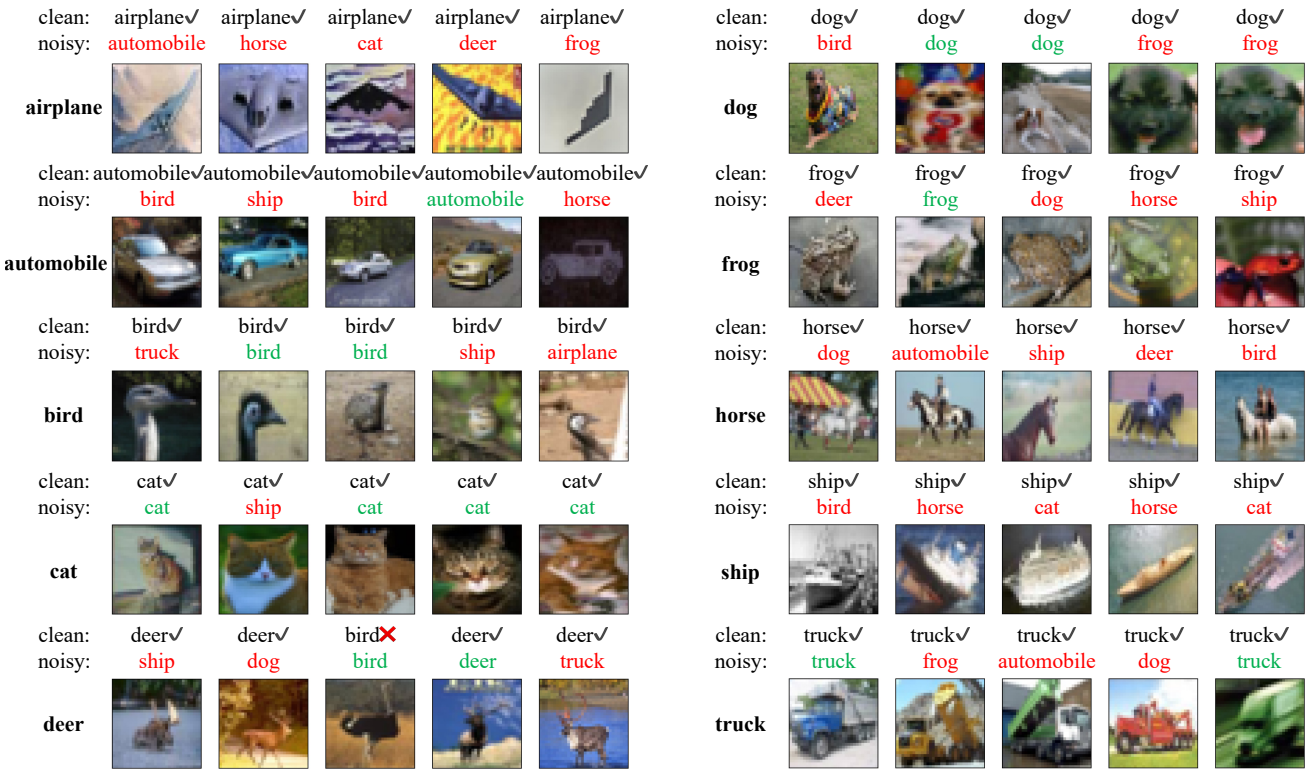



Figure 2. Visualization of images and labels in CIFAR-10 dataset with 90% label noise. For each class, we select and show those images with top-5 largest cosine similarity features to the prototypes. Most of the image features are correctly assigned to the correct class, which demonstrates the effectiveness of our proposed PLR loss.

Table 3. Training time (hours) on CIFAR-10 dataset with 80% symmetric noise on RTX 4090

| DivideMix | C2D | ScanMix | PLReMix (Ours) |
|---|---|---|---|
| 3.4h | 4.5h+3.4h | 4.5h+0.4h+6.5h | 6.0h |

## C. Visualization

In Fig. 1, we use t-SNE [7] to visualize the features of training images for different noise modes and ratios. We randomly select 5% of samples from each class. Circles represent the features $\boldsymbol{q}$ and stars represent the class prototypes $\boldsymbol{P}$. It can be seen that the feature embeddings form distinct clusters based on their latent ground truth labels rather than the given noisy labels, which demonstrates the robustness of the proposed method.

In Fig. 2, we visualize the images that have the top-5 largest cosine similarity features to the prototypes of each class on the CIFAR-10 dataset with 90% symmetric noise. Corresponding ground truth labels and given noisy labels are listed above each image. If the noisy label differs from the latent ground truth, it is colored in red; otherwise in green. We use a ✓ to indicate that an image has been correctly assigned to its corresponding cluster and × if not. As can be seen, most images have been assigned to its ground truth cluster, which shows the effectiveness of our method.

## D. Training Time Analysis

In Tab. 3, we compare the training time of our proposed PLReMix framework with several methods on CIFAR-10 with 80% symmetric noise, using a single RTX 4090 GPU. Our PLReMix is slower than DivideMix [4] as an auxiliary PLR loss is added, but is faster than C2D [8] and ScanMix [5], both of which utilize a SimCLR pre-trained model weights.

## References

[1] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Proceedings of the European Conference on Computer Vision*, pages 630–645. Springer, 2016. 1

[4] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2019. 1, 3, 4

[5] Ragav Sachdeva, Filipe Rolim Cordeiro, Vasileios Belagiannis, Ian Reid, and Gustavo Carneiro. Scanmix: Learning from severe label noise via semantic clustering and semi-supervised learning. *Pattern Recognition*, 134:109121, 2023. 4

[6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017. 1

[7] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 4

[8] Evgenii Zheltonozhskii, Chaim Baskin, Avi Mendelson, Alex M Bronstein, and Or Litany. Contrast to divide: Self-supervised pre-training for learning with noisy labels. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1657–1667, 2022. 4