CrossSplit: Mitigating Label Noise Memorization through Data Splitting

Jihye Kim [1 2] **Aristide Baratin** [3] **Yan Zhang** [3] **Simon Lacoste-Julien** [3 4 5]

Abstract

We approach the problem of improving robustness of deep learning algorithms in the presence of label noise. Building upon existing label correction and co-teaching methods, we propose a novel training procedure to mitigate the memorization of noisy labels, called CrossSplit, which uses a pair of neural networks trained on two disjoint parts of the labeled dataset. CrossSplit combines two main ingredients: (i) Cross-split label correction. The idea is that, since the model trained on one part of the data cannot memorize example-label pairs from the other part, the training labels presented to each network can be smoothly adjusted by using the predictions of its peer network; (ii) Cross-split semi-supervised training. A network trained on one part of the data also uses the unlabeled inputs of the other part. Extensive experiments on CIFAR-10, CIFAR-100, Tiny-ImageNet and mini-WebVision datasets demonstrate that our method can outperform the current state-of-the-art in a wide range of noise ratios. The project page is at https://rlawlgul.github.io/.

1. Introduction

A large part of the success of deep learning algorithms relies on the availability of massive amounts of labeled data, via e.g. web crawling (Li et al., 2017a) or crowd-sourcing platforms (Song et al., 2019). However, while these data-collection methods enable to bypass cost-prohibitive human annotations, they inherently yield a lot of mislabeled samples (Xiao et al., 2015; Li et al., 2017a). This leads to a degradation of the performance, especially considering that deep neural networks have enough capacity to fully memorize noisy labels (Zhang et al., 2017; Liu et al., 2020; Arpit et al., 2017). An important issue in the field is therefore to adapt the training process to improve robustness under label noise.

This problem has been addressed in various ways in the recent literature. Two common approaches are *label correction* and *sample selection*. The first one focuses on correcting the noisy labels during training, e.g. by using soft labels defined as convex combinations of the assigned label and the model prediction (Reed et al., 2015; Arazo et al., 2019; Lu & He, 2022). Another common approach uses sample selection mechanisms, which separate clean examples from noisy ones during training (Li et al., 2020; Karim et al., 2022; Han et al., 2018; Yu et al., 2019), e.g. using a small-loss criterion (Li et al., 2019). Current state-of-the-art methods (Li et al., 2020; Karim et al., 2022) combine epoch-wise sample selection with a co-teaching procedure (Han et al., 2018; Yu et al., 2019) where two networks are trained simultaneously, each of them using the sample selection of the other so as to mitigate confirmation bias. Semi-supervised learning (SSL) techniques are then used where the selected noisy examples are treated as unlabeled data.

Despite the popularity and success of these methods, they are not exempt from drawbacks. Existing label correction methods define soft target labels in terms of their own prediction, which may become unreliable as training progresses and memorization occurs (Lu & He, 2022). Sample selection procedures rely on criteria to filter out noisy examples which are subject to selection errors – in fact, making an accurate distinction between mislabeled and inherently difficult examples is a notoriously challenging problem (D'souza et al., 2021; Pleiss et al., 2020; Baldock et al., 2021).

The goal of this paper is to propose a novel robust training scheme that addresses some of these drawbacks. The idea is to bypass the sample selection process by using a random splitting of the data into two disjoints parts, and to train a separate network on each of these splits. The rationale is that the model trained on one part of the data cannot memorize input-label pairs from the other part. We propose to correct the labels presented to each network by using a combination of the assigned label and the prediction of the peer network. This procedure allows us to avoid the memorization of examples without significantly degrading the learning of difficult examples. Cross-split semi-supervised

[1]Samsung Advanced Institute of Technology (SAIT), Suwon, South Korea [2]Work done as a visiting researcher at SAIT AI Lab, Montreal, Canada [3]SAIT AI Lab, Montreal, Canada [4]Mila, Université de Montreal, Canada [5]Canada CIFAR AI Chair. Correspondence to: Jihye Kim <jihye32.kim@samsung.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

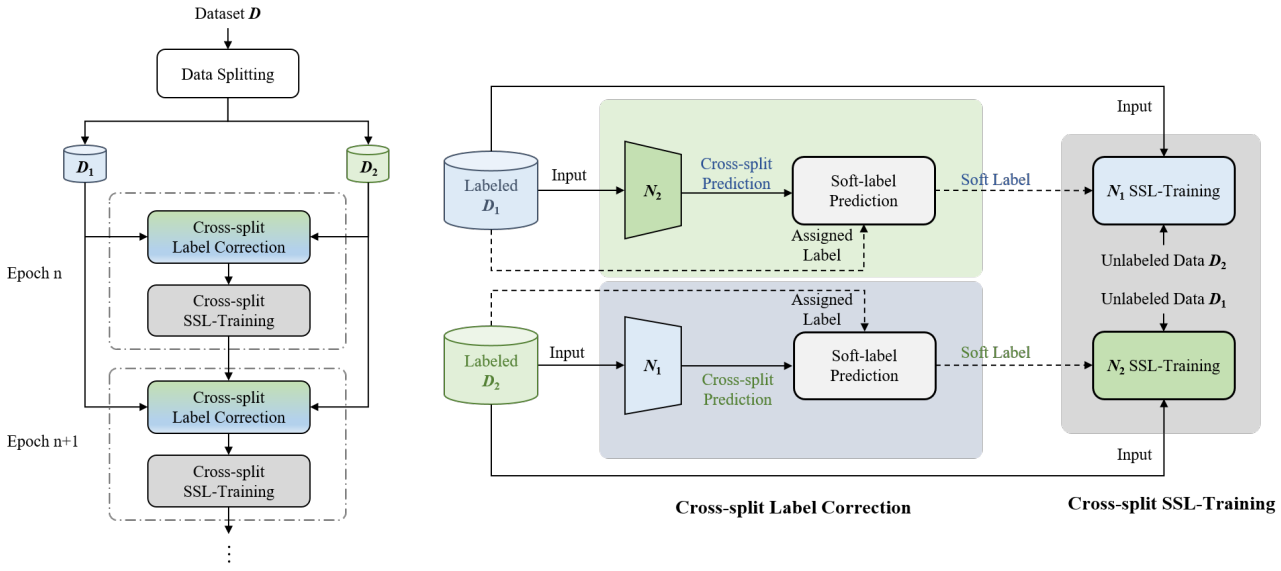

Figure 1. CrossSplit splits the original training labelled dataset into two disjoint parts and trains a separate network on each of these splits. The dataset each network is trained on is also used by the peer network as unlabeled data for semi-supervised learning (SSL). At each training epoch, *CrossSplit* uses a cross-split label correction scheme that defines soft labels in terms of the peer prediction.

learning is then performed where the data each network is trained on is also used as unlabeled data by the peer network.

Our contributions are summarized as follows:

- We introduce *CrossSplit* for robust training (Section 2, overview in Figure 1). CrossSplit departs from existing methods by using a pair of networks trained on *two random splits* of the labeled dataset, leading to a novel *label correction procedure* based on peer-predictions and a cross-split semi-supervised training process.
- Through experimental analysis, we verify that this data splitting and training scheme help in reducing the memorization of noisy labels (Figure 2), which in turn improves robustness under label noise.
- Through extensive experiments on CIFAR-10, CIFAR-100, Tiny-ImageNet, and mini-WebVision datasets, we show that our method can outperform the current state-of-the-art in a wide range of noise ratios (Section 4).
- We perform a thorough ablation study of the different components of our procedure (Section 4.6).

2. Proposed Method

In this section we introduce *CrossSplit* for alleviating memorization of noisy labels in order to improve robustness.

Setup Just like in standard co-training (Blum & Mitchell, 1998) and co-teaching (Han et al., 2018; Yu et al., 2019) schemes, *CrossSplit* simultaneously trains two neural networks $\mathcal{N}_1$ and $\mathcal{N}_2$. While these networks can in principle be completely different models, for simplicity we use the same architecture with two distinct sets of parameters. Our procedure begins with a random splitting of the labeled dataset $\mathcal{D}$ into two disjoint subsets $\mathcal{D}_1$ and $\mathcal{D}_2$ of equal size. At each training epoch, *CrossSplit* includes a label correction step where the labels presented to each network are corrected using the peer network prediction. This is a simple yet effective way to mitigate memorization of the noisy labels, since each network cannot memorize the input-label pairs

Algorithm 1 CrossSplit: Cross-split SSL training based on cross-split label correction

Input: Split training set $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2\}$, pair of networks $\mathcal{N}_1, \mathcal{N}_2$, warmup epoch E_{warm}, total number of epochs E_{max}.
$\theta_1, \theta_2 \leftarrow$ Initialize network parameters
$\theta_1, \theta_2 \leftarrow$ Warmup supervised training on whole dataset for E_{warm} epochs
for *epoch* $\in [E_{warm} + 1, \ldots, E_{max}]$ **do**
  1. Training $\mathcal{N}_1$:
  1.1: Perform cross-split label correction Equation (1) for labeled $\mathcal{D}_1$ using the predictions of $\mathcal{N}_2$ (see Section 2.1).
  1.2: Perform SSL training (Sohn et al., 2020; Zhang et al., 2018) using (soft)-labeled $\mathcal{D}_1$ as labeled data and $\mathcal{D}_2$ as unlabeled data (see Section 2.2).
  2. Analogous training for $\mathcal{N}_2$.
end
Return: θ_1, θ_2.

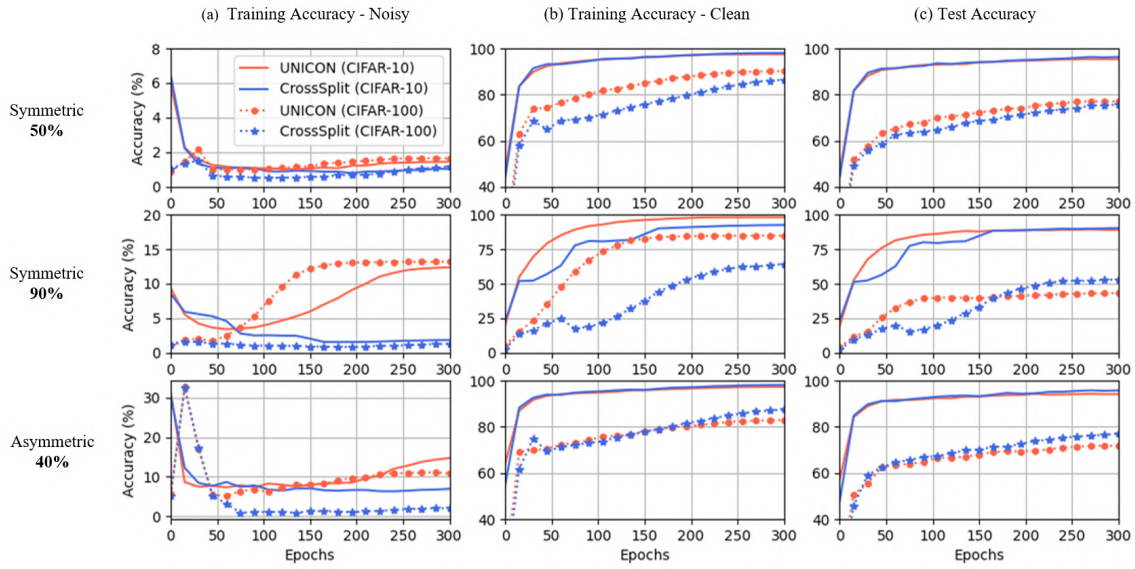

Figure 2. Memorization of clean and noisy training samples of CIFAR-10 and CIFAR-100 for different types of noise and noise ratios. Compared to UNICON (Karim et al., 2022), *CrossSplit* induces less memorization (lower accuracy) on the noisy labels while having comparable accuracy on clean samples. It is interesting to note that in the case of a very high noise ratio (90%), *CrossSplit* has a lower training accuracy on clean data than UNICON, yet yields a higher test performance. This shows how important reducing memorization is, since the lower memorization of noisy labels completely offsets the lower accuracy on clean samples.

presented to its peer. Following (Li et al., 2020; Karim et al., 2022), *CrossSplit* then leverages semi-supervised learning techniques; the novelty here is to bypass the usual sample selection of noisy data, and to rely instead on a mere cross-split training: $\mathcal{N}_1$ is trained on $\mathcal{D}_1$ (with soft labels) and uses the inputs of $\mathcal{D}_2$ as unlabeled data; $\mathcal{N}_2$ is trained on $\mathcal{D}_2$ (with soft labels) and uses the inputs of $\mathcal{D}_1$ as unlabeled data. The training procedure is illustrated in Figure 1 and summarized in Algorithm 1. For the final predictions, we use the same way to combine the predictions of the two models ($\mathcal{N}_1$ and $\mathcal{N}_2$) as prior works like DivideMix (Li et al., 2020) and UNICON (Karim et al., 2022): we obtain the final prediction by summing the two logit vectors from the two models.

We provide below a more detailed description of the different components of *CrossSplit*.

2.1. Cross-split Label Correction

Label correction serves the important purpose of identifying which examples are likely to be mislabeled. At every epoch of our training procedure, for each of the two networks, we will use soft labels defined as convex combinations of the assigned label and the peer network prediction. The crucial aspect is that due to the data splitting, the peer network cannot memorize the label that it is modifying. This is in contrast to existing methods (Reed et al., 2015; Li et al., 2020; Karim et al., 2022; Lu & He, 2022) that combine assigned labels with the network's own prediction: if the network has memorized the noisy label, it simply reinforces the mislabeling.

Consider the network $\mathcal{N}_1$ and let $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_1$, where $\mathbf{x}_i$ is an input image and $\mathbf{y}_i$ is the one-hot vector associated to its (possibly noisy) class label. We define the soft label $\mathbf{s}_i$ as the following convex combination of $\mathbf{y}_i$ and the cross-split probability (softmax) vector, $\hat{\mathbf{y}}_{\text{peer},i} = \mathcal{N}_2(\mathbf{x}_i)$:

$$\mathbf{s}_i = \beta_i \hat{\mathbf{y}}_{\text{peer},i} + (1 - \beta_i)\mathbf{y}_i \tag{1}$$

$$\beta_i = \gamma(\text{JSD}_{\text{norm}}(\hat{\mathbf{y}}_{\text{peer},i}, \mathbf{y}_i) - 0.5) + 0.5 \tag{2}$$

where JSD_{norm} is a normalized version of the Jensen-Shannon Divergence (JSD) described in Equation (4) below, and γ is a relaxation parameter.[1] Intuitively, when the peer network confidently predicts the assigned label $\mathbf{y}_i$, β_i is small and Equation (1) picks a soft label that is close to $\mathbf{y}_i$. For a confident peer prediction that disagrees with $\mathbf{y}_i$, the soft label shifts towards the cross-prediction label $\hat{\mathbf{y}}_{\text{peer},i}$. In practice, "confident prediction" simply refers to the distance between the peer prediction and the label being close to 0 or 1 (as measured by the JSD).

To understand the role of β_i in Equation (1), it is important to keep in mind that we are in the noisy label setting, so labels

[1] This parameter enables us to control the range of β_i, especially at the beginning of training where we may expect the JSD values to be noisy. We explain this in more detail in Appendix A.1.

are frequently wrong. When the peer network's prediction departs from the original label (large JSD), then we assume that it successfully generalized and is more trustworthy than a possibly wrong label. Our empirical results show that this is a useful assumption to make (see Table 4 and Table 5). Yet, if we always rely purely on the peer prediction, then neither model is ever trained on any labels, so neither network is learning anything useful. Especially at the start of training, the peer network should not be relied upon too much. Incorporating the label through β_i is thus crucial to learning.

Class-balancing coefficient normalization UNICON (Karim et al., 2022) noted that when performing sample selection, the selection threshold should vary between different classes. Otherwise, the model is biased towards selecting samples from easy classes to be clean, while rejecting clean samples from harder classes as noisy. We can adapt this idea to our framework by thinking of the weighting from β_i as "soft" sample selection. In particular, we normalize the standard JSD that (Karim et al., 2022) use in such a way that, *within each class*, it ranges from 0 to 1.

To compute this, we keep track of the minimum and maximum JSD values within each class, which we compute at the beginning of every epoch. For each class, encoded by the one-hot vector $\mathbf{y}$, we thus compute the quantities

$$\begin{aligned} \text{JSD}_{\mathbf{y}}^{\min} &:= \min_{\{j|\mathbf{y}_j=\mathbf{y}\}} \text{JSD}(\hat{\mathbf{y}}_{\text{peer},j}, \mathbf{y}), \\ \text{JSD}_{\mathbf{y}}^{\max} &:= \max_{\{j|\mathbf{y}_j=\mathbf{y}\}} \text{JSD}(\hat{\mathbf{y}}_{\text{peer},j}, \mathbf{y}). \end{aligned} \tag{3}$$

For each example, we then normalize the JSD through shifting and scaling, using the values (Equation (3)) associated to its class.

$$\text{JSD}_{\text{norm}}(\hat{\mathbf{y}}_{\text{peer},i}, \mathbf{y}_i) := \frac{\text{JSD}(\hat{\mathbf{y}}_{\text{peer},i}, \mathbf{y}_i) - \text{JSD}_{\mathbf{y}_i}^{\min}}{\text{JSD}_{\mathbf{y}_i}^{\max} - \text{JSD}_{\mathbf{y}_i}^{\min}} \tag{4}$$

2.2. Cross-split SSL-Training

The two neural networks $\mathcal{N}_1$ and $\mathcal{N}_2$ are each trained on only half the amount of labeled data, which can degrade performance. We thus look towards semi-supervised learning, which lets us train $\mathcal{N}_1$ using *unlabeled* data (to avoid memorization) from $\mathcal{D}_2$ and $\mathcal{N}_2$ using unlabeled data from $\mathcal{D}_1$.

We use a cross-split semi-supervised training procedure. At each training epoch, $\mathcal{N}_1$ is trained on (soft)-labeled $\mathcal{D}_1$ with the unlabeled samples from $\mathcal{D}_2$ and $\mathcal{N}_2$ is trained on (soft)-labeled $\mathcal{D}_2$ with the unlabeled samples from $\mathcal{D}_1$ (see Figure 1). Regarding the specific techniques used, we reproduce the main ingredients of existing methods (Li et al., 2020; Karim et al., 2022), by following FixMatch (Sohn et al., 2020) and applying MixUp (Zhang et al., 2018) augmentation. Just like UNICON (Karim et al., 2022), the semi-supervised loss is combined with a contrastive loss evaluated on the unlabeled dataset to further mitigate noisy label memorization. The final loss is expressed as

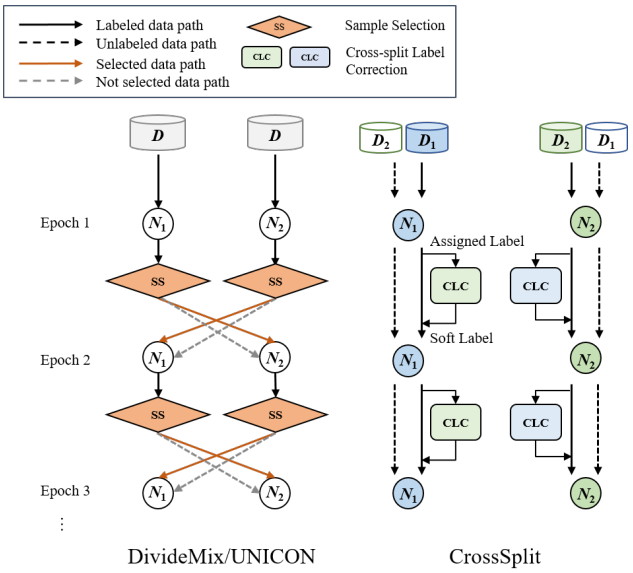

Figure 3. Comparison of the DivideMix (Li et al., 2020), UNICON (Karim et al., 2022), and *CrossSplit* co-teaching pipelines. The data flow is represented with solid lines for labeled data and dotted lines for unlabeled data. All three methods train two networks ($\mathcal{N}_1$ & $\mathcal{N}_2$) simultaneously. In DivideMix and UNICON, at every epoch, each network separates clean samples (orange solid line) and noisy samples (gray dotted line) using a small loss criterion, and transfers the two subsets to its peer network for subsequent semi-supervised learning. By contrast, *CrossSplit* splits the original training dataset into two halves and trains each network on one of these splits. For each of the two networks, we use soft labels defined as convex combinations of the assigned label and the peer network prediction via cross-split label correction (CLC) process. The data each network is trained on is also used by the peer network as unlabeled data for semi-supervised learning.

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{semi}} + \lambda_c \mathcal{L}_{\text{contrastive}}, \tag{5}$$

where λ_c is contrastive loss coefficient. The effect of the contrastive loss is provided in Appendix B.2.

3. Related Work

The problem of learning with noisy labels has been approached in various ways in the literature. These include label correction (Reed et al., 2015; Arazo et al., 2019; Zhang et al., 2020; Li et al., 2020; Lu & He, 2022), noise robust loss (Zhang & Sabuncu, 2018; Ma et al., 2020), loss correction (Goldberger & Ben-Reuven, 2017) and sample selection (Li et al., 2020; Karim et al., 2022; Han et al., 2018; Yu et al., 2019) based methods. Most relevant to our work are *label*

correction and *sample selection*, which we discuss now in more detail.

Label correction methods In order to mitigate the negative influence of noisy labels in training, some works have focused on gradually adjusting the assigned label based on the model's prediction (Reed et al., 2015; Arazo et al., 2019; Zhang et al., 2020; Li et al., 2020; Lu & He, 2022; Ma et al., 2018; Tanaka et al., 2018b). *Bootstrapping* (Reed et al., 2015) generates new regression targets by combining the assigned label and the model's prediction, using the same fixed combination weight for all samples. *M-correction* (Arazo et al., 2019) uses instead dynamic weights defined in terms of the sample's training loss values. Follow-up works proposed to incorporate the prediction confidence or use ensemble predictions by the exponential moving average over multiple previous epochs in the design of the weights (Zhang et al., 2020; Lu & He, 2022). However, existing label correction methods have a limitation in that labels are corrected only based on their own prediction. This can lead to the memorization of noisy samples which only reinforces their own mislabelings. This is in contrast to the label correction method proposed in our work, which generates soft labels as combinations of the assigned label and the peer network prediction; the peer network cannot memorize the label because it never sees that label during its own training.

Sample selection-based methods Another common approach is to identify the noisy samples, e.g., using a small-loss criterion (Li et al., 2019), to separate them from the clean ones, and to use the two subsets of samples in a different way during training (Han et al., 2018; Li et al., 2020; Karim et al., 2022). The selected clean set is typically used for conventional supervised learning; the noisy samples are either excluded from training (Han et al., 2018) or treated as unlabeled data for semi-supervised learning (Li et al., 2020; Karim et al., 2022).

Despite the empirical success of this approach, it has some limitations. Sample selection processes require hyper-parameter setting for selection (e.g., noise ratio, threshold value for selection); how well this selection is done affects performance. They also face the difficult challenge to distinguish between mislabeled data, which should not be memorized, and difficult examples whose labels nevertheless carry useful information (Feldman, 2020). Our work proposes a method that bypasses the sample selection process, where the hard decision as to whether a sample is clean or not is replaced by a soft label correction using the peer network.

Co-training methods State-of-the-art sample selection methods take advantage of training two models simultaneously in order to prevent confirmation bias (Li et al., 2020; Karim et al., 2022). One network selects its small-loss samples (considered as clean samples) to teach its peer network for subsequent training (Han et al., 2018; Yu et al., 2019). This idea of network cooperation can be traced back to co-training (Blum & Mitchell, 1998), which can be shown to improve the performance of learning by unlabeled data in semi-supervised learning. In the original version (Blum & Mitchell, 1998), multiple classifiers are trained on distinct views of the data, e.g., mutually exclusive feature sets for the same example, and exchange their predictions. For example, data with high confidence prediction can be added for re-training to the data seen by the peer model (Ma et al., 2017). Our approach is similar in spirit, but instead of working with different views of the data, we train different models on disjoint subsets of the dataset. Figure 3 illustrates the differences between *CrossSplit* and several other co-teaching schemes used in the recent literature (Han et al., 2018; Li et al., 2020; Karim et al., 2022).

4. Experiments

4.1. Datasets

We conduct experiments both on datasets with simulated label noise and datasets with natural label noise. Simulating the noise allows us to control the noise level, analyze the memorization behavior of our algorithm and test a variety of scenarios. On the other hand, working with naturally noisy datasets enables practical evaluation in situations where the type and level of noise are unknown.

CIFAR-10/100 datasets (Krizhevsky et al., 2009) each contains 50K training and 10K testing 32×32 coloured images. Following the setup of previous works (Li et al., 2017b; Tanaka et al., 2018a; Yu et al., 2019; Li et al., 2020; Karim et al., 2022), we use both symmetric and asymmetric label noise. Symmetric label noise is generated by re-assigning to a portion of the training data in each class, a label chosen uniformly at random among all other classes. Asymmetric label noise mimics real-world label noise more closely: the labels are chosen among similar classes (e.g., Bird $\rightarrow$ Airplane, Deer $\rightarrow$ Horse, Cat $\rightarrow$ Dog). For CIFAR-100, labels are flipped circularly within the super-classes. We simulate a wide range of noise levels: 0% of label noise, 20% - 90% for symmetric label noise and 10% - 40% for asymmetric label noise.

Tiny-ImageNet (Le & Yang, 2015) is a subset of the ImageNet dataset with 100K 64×64 coloured images distributed within 200 classes. Each class has 500 training images, 50 test images and 50 validation images. We experiment on Tiny-ImageNet with simulated symmetric label noise.

Mini-WebVision (Li et al., 2017a) contains 2.4 million images from websites Google and Flicker and contains many naturally noisy labels. The images are categorized into 1,000 classes and following (Karim et al., 2022), we use the top-50 classes from the Google images of WebVision for training.

4.2. Experimental details

Architectures For CIFAR-10, CIFAR-100 and Tiny-ImageNet, in line with (Li et al., 2020; Karim et al., 2022), we use a PreAct ResNet18 (He et al., 2016a) architecture. For mini-WebVision, we use ResNet18. We give training details in Appendix A.2.

4.3. Results

In this section, we compare the performance of *CrossSplit* with existing methods (Section 4.3.1), which include label correction and sample-selection methods. We also analyze the memorization behaviour of the algorithm (Section 4.5). Our baselines are Bootstrapping (Reed et al., 2015), JPL (Kim et al., 2021), M-Correction (Arazo et al., 2019), MOIT (Ortego et al., 2021), SELC (Lu & He, 2022), Sel-CL(Li et al., 2022), DivideMix (Li et al., 2020), ELR (Liu et al., 2020), and UNICON (Karim et al., 2022).

4.3.1. Performance

Table 1 shows test accuracies on CIFAR-10 and CIFAR-100 with different levels of noise ratios ranging from 20% to 90% for symmetric noise and 10% to 40% for asymmetric noise respectively. We observe that *CrossSplit* consistently outperforms the competing baselines under a wide range of noise levels for the two types of noise models. In particular, we note a large performance improvement in the case of asymmetric label noise (which is more likely to occur in real scenarios) for both CIFAR-10 and CIFAR-100. Even for symmetric label noise, we see performance improvements in all cases except for CIFAR-100 with a 50% noise ratio. Additionally, we show visual comparisons of the features learned by UNICON (Karim et al., 2022) and *CrossSplit* in Appendix B.5. These show that the representations learned by our model are more distinct between classes, particularly when the noise is high (see Figure B.3). Most of the recent existing methods (Li et al., 2020; Karim et al., 2022) have been trained with PreAct ResNet-18 (PRN-18) (He et al., 2016b). In order to analyze the effect of model size on performance, we additionally conduct the same experiments with a deeper network architecture, PreAct ResNet-34 (PRN-34). In most cases, the bigger capacity of PRN-34 helps in improving the overall performance. However, in cases of extremely high noise (90% for symmetric noise on CIFAR-10 and CIFAR-100), its accuracies are lower than PRN-18 (90% for symmetric noise on CIFAR-10: 85.3% (vs. PRN-18: 91.3%)) or not stable with having a relatively larger standard deviation (3.61% in CIFAR-10 and 3.43% in CIFAR-100) than PRN-18 (0.79% on CIFAR-10 and 1.78% on CIFAR-100). It may be due to a relatively larger possibility of memorization of noisy labels under extreme noise ratio. For reference, we also show the results without label noise (0%). Compared with CE-based learning (He et al., 2016b) and MixUp (Zhang et al., 2018) under no label noise, *CrossSplit* has better performance both on CIFAR-10 and CIFAR-100. The results of MixUp are imported from (Zhang et al., 2018) (PRN-18).

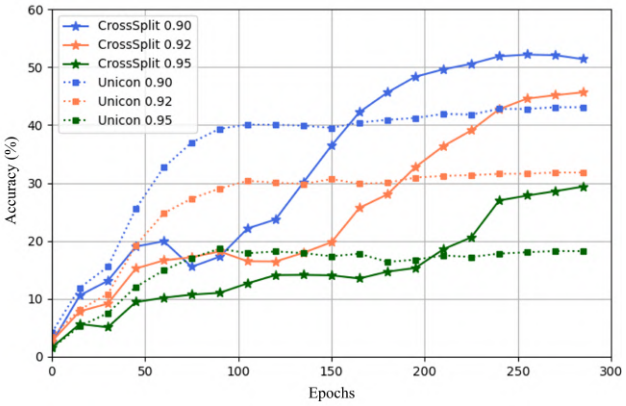

Figure 4. Comparisons of test accuracy (%) of *CrossSplit* and UNICON under extreme label noise on CIFAR-100. While learning progresses slower for *CrossSplit* at the beginning (possibly due to the untrained peer network not effectively correcting labels yet), the final performance is consistently superior to UNICON.

For the Tiny-ImageNet dataset, we model symmetric label noise with two noise ratios, 20% and 50%. Table 2 shows test results both with the highest (Best) and the average over the last 10 epochs (Avg.). In this case, compared to existing algorithms, we observe a slight degradation of performance for a 50% noise ratio with respect to the best competing baseline, and similar performance for a 20% noise ratio. Our results here are largely similar to the state-of-the-art.

Table 3 shows performance comparisons for the mini-WebVision dataset, which is the most realistic task setting because the noise is present naturally due to the web-crawled nature of the data; the noise levels and structure of the noise are unknown. There is a 0.88 % improvement over the current state-of-the-art UNICON (Karim et al., 2022), which demonstrates the benefits of our model in this experiment setting closest to the real world.

4.4. Additional results under extreme label-noise

(Karim et al., 2022) show excellent performance of the current state-of-the-art UNICON even in the case of extremely high levels of label noise (over 90%). Here we provide analogous results for *CrossSplit* under extreme noise ratio (90%,

Table 1. Test accuracy (%) comparison on CIFAR-10 (left) and CIFAR-100 (right) without label noise and with symmetric and asymmetric label noise. Our model achieves state-of-the-art performance on almost every dataset-noise combination. The best scores are **boldfaced**, and the second best ones are underlined. The baseline results are imported from (Karim et al., 2022; Li et al., 2020; 2022). For CrossSplit, mean and standard deviation of best accuracy are calculated over 3 repetitions of the experiments. The results are sorted according to their performance in the case of a 20% symmetric noise ratio.

	CIFAR-10								CIFAR-100							
Noise type	-	Symmetric				Asymmetric			-	Symmetric				Asymmetric		
Method/Noise ratio	0%	20%	50%	80%	90%	10%	30%	40%	0%	20%	50%	80%	90%	10%	30%	40%
CE	95.4	86.8	79.4	62.9	42.7	88.8	81.7	76.1	77.3	62.0	46.7	19.9	10.1	68.1	53.3	44.5
Bootstrapping (Reed et al., 2015)	-	86.8	79.8	63.3	42.9	-	-	-	-	62.1	46.6	19.9	10.2	-	-	-
JPL (Kim et al., 2021)	-	93.5	90.2	35.7	23.4	94.2	92.5	90.7	-	70.9	67.7	17.8	12.8	72.0	68.1	59.5
M-Correction (Arazo et al., 2019)	-	94.0	92.0	86.8	69.1	89.6	92.2	91.2	-	73.9	66.1	48.2	24.3	67.1	58.6	47.4
MOIT (Ortego et al., 2021)	-	94.1	91.1	75.8	70.1	94.2	94.1	93.2	-	75.9	70.1	51.4	24.5	77.4	75.1	74.0
SELC (Lu & He, 2022)	-	95.0	-	78.6	-	-	-	92.9	-	76.4	-	37.2	-	-	-	73.6
Sel-CL (Li et al., 2022)	-	95.5	93.9	89.2	81.9	95.6	95.2	93.4	-	76.5	72.4	59.6	48.8	78.7	76.4	74.2
MixUp (Zhang et al., 2018)	95.8	95.6	87.1	71.6	52.2	93.3	83.3	77.7	78.9	67.8	57.3	30.8	14.6	72.4	57.6	48.1
ELR (Liu et al., 2020)	-	95.8	94.8	93.3	78.7	95.4	94.7	93.0	-	77.6	73.6	60.8	33.4	77.3	74.6	73.2
UNICON (Karim et al., 2022)	-	96.0	95.6	93.9	90.8	95.3	94.8	94.1	-	78.9	**77.6**	63.9	44.8	78.2	75.6	74.8
DivideMix (Li et al., 2020)	-	96.1	94.6	93.2	76.0	93.8	92.5	91.7	-	77.3	74.6	60.2	31.5	71.6	69.5	55.1
CrossSplit (PRN-18)	**97.0**	**96.9**	**96.3**	**95.4**	**91.3**	**96.9**	**96.4**	**96.0**	**81.7**	**79.9**	75.7	**64.6**	**52.4**	**80.7**	**78.5**	**76.8**
	±0.16	±0.05	±0.05	±0.64	±0.79	±0.04	±0.16	±0.12	±0.25	±0.19	±0.18	±1.43	±1.78	±0.05	±0.19	±0.66
CrossSplit (PRN-34)	**97.3**	**97.1**	**96.5**	**95.2**	85.3	**97.2**	**96.6**	**96.1**	**83.0**	**81.4**	77.2	**67.0**	**52.6**	**82.6**	**80.5**	**79.1**
	±0.16	±0.16	±0.24	±0.59	±3.61	±0.09	±0.11	±0.08	±0.15	±0.38	±0.25	±0.49	±3.43	±0.15	±0.27	±0.40

92%, and 95%). Table 6 shows the results for CIFAR-100 with symmetric label noise. The performance of UNICON (except for label noise of 90%) is obtained by re-running their publicly available code[2]. In Figure 4, at the early training epochs, the performance of *CrossSplit* (star marked solid line) may seem inferior compared to UNICON (square marked dashed line). This can be interpreted as the fact that some noisy labels are likely to be temporarily included during training due to the lack of a selection mechanism. However, as training proceeds, the effect of noisy labels is gradually minimized by our cross-split label correction process, so it can be confirmed that the performance improves rapidly at later training epochs and consistently at all noise levels. We observe that *CrossSplit* outperforms UNICON for all noise levels on CIFAR-100 (see Table 6 and Figure 4).

4.5. Memorization analysis

The previously-discussed results show that *CrossSplit* compares well with – and often outperforms – the competing baselines. This begs the question of the origin of this performance gap. The core hypothesis of the paper is that our method induces an implicit regularization that better prevents the memorization of noisy labels. In this section, we investigate this hypothesis by quantifying this memorization and comparing it with the current state-of-the-art UNICON (Karim et al., 2022).

To do so, we check the training accuracy separately on the clean and noisy samples of CIFAR-10 and CIFAR-100 with different noise types and ratios (symmetric-50%, 90% and asymmetric-40% noise). The results are shown in Figure 2. From left to right, the plots show (a) the training accuracy for noisy (mislabeled) samples, (b) the training accuracy for clean samples, and (c) the test accuracy.

Discussion During the initial warm-up period where the whole dataset is used for training, we observe that the noisy samples are increasingly memorized, especially on CIFAR-100 (Figure 2). Immediately after the warm-up period though, some forgetting often occurs for both methods, i.e., the accuracy on noisy samples tends to decrease. However, in the case of UNICON, memorization rises again within a few epochs. By contrast, *CrossSplit* manifestly continues to mitigate this memorization while maintaining the fit of clean samples (Figure 2 (b)). This effect seems to correlate with the gain of performance observed in Figure 2 (c). In summary, we find that *CrossSplit* effectively reduces memorization of noisy labels in contrast to UNICON, which explains its superior performance.

4.6. Ablation Study

In this section, we perform an ablation study to demonstrate the effectiveness of some key components of CrossSplit: data splitting, class-balancing coefficient normalization by JSD_{norm} (Equation (4)), and cross-split label correction. We remove each component to quantify its contribution to the overall performance on CIFAR-10/100 with symmetric-50% and 90% noise and CIFAR-10/100 with asymmetric-10%

[2] https://github.com/nazmul-karim170/UNICON-Noisy-Label

Tables 2 & 3. Test accuracy (%) comparison on Tiny-ImageNet (left) and mini-WebVision (right). Our model is competitive with the state-of-the-art (only small differences in performance) on Tiny-ImageNet with artificial noise, and surpasses the state-of-the-art on mini-Webvision with real-world noise. The best scores are **boldfaced**, and the second best ones are underlined. In Table 2, Best and Avg. mean highest and average accuracy over the last 10 epochs. The baseline results are imported from (Karim et al., 2022) and sorted according to their best performance in the case of a 20% noise ratio. In Table 3, the baseline results are sorted by best performance.

Table 2. Tiny-ImageNet

Noise type	Symmetric			
Noise ratio	20%		50%	
Method	Best	Avg.	Best	Avg.
CE	35.8	35.6	19.8	19.6
Decoupling (Malach & Shalev-Shwartz, 2017)	37.0	36.3	22.8	22.6
MentorNet (Jiang et al., 2018)	45.7	45.5	35.8	35.5
Co-teaching+ (Yu et al., 2019)	48.2	47.7	41.8	41.2
M-Correction (Arazo et al., 2019)	57.2	56.6	51.6	51.3
NCT (Sarfraz et al., 2021)	58.0	57.2	47.8	47.4
UNICON (Karim et al., 2022)	**59.2**	58.4	**52.7**	**52.4**
CrossSplit (ours)	59.1	**58.8**	52.4	52.0

Table 3. Mini-WebVision

Method	Best	Last
Decoupling (Malach & Shalev-Shwartz, 2017)	62.54	-
MentorNet (Jiang et al., 2018)	63.00	-
Co-teaching (Han et al., 2018)	63.58	-
Iterative-CV (Chen et al., 2019)	65.24	-
ELR (Liu et al., 2020)	73.00	71.88
SELC (Lu & He, 2022)	74.38	-
MixUp (Zhang et al., 2018)	74.96	73.76
DivideMix (Li et al., 2020)	76.08	74.64
UNICON(Karim et al., 2022)	77.60	-
CrossSplit (ours)	**78.48**	**78.07**

Table 4. Ablation study on CIFAR-10: Test accuracy (%) of different setting on CIFAR-10 with varying noise rates (50% - 90% for Symmetric and 10% - 40% for Asymmetric noise). We see that there is a minor difference when removing class-balancing normalization with lower noise ratios, but a large degradation in performance if it is removed for high noise ratios. Mean and standard deviation of best and average of last 10 epochs are calculated over 3 repetitions of the experiments. The best results are highlighted in **boldfaced** and scores that differ from them by more than 5% are marked in red.

Noise type	Symmetric				Asymmetric			
Noise ratio	50%		90%		10%		40%	
Method	Best	Last	Best	Last	Best	Last	Best	Last
CrossSplit	96.34±0.05	96.23±0.07	**91.25**±0.79	**91.02**±0.77	96.85±0.04	96.74±0.07	96.01±0.12	95.88±0.13
w/o data splitting	96.10±0.04	95.96±0.00	90.30±0.13	89.93±0.24	96.76±0.05	96.63±0.06	92.16±0.09	86.24±0.37
w/o class-balancing normalization	**96.73**±0.13	**96.61**±0.07	75.54±2.82	74.88±2.50	**97.33**±0.02	**97.20**±0.02	**96.22**±0.07	**96.04**±0.12
w/o cross-split label correction	96.12±0.05	95.99±0.03	90.83±0.25	90.08±0.40	**97.33**±0.08	97.15±0.09	96.12±0.14	95.95±0.10

and 40% noise. Table 4 and Table 5 show the test accuracy in different ablation settings for CIFAR-10 and CIFAR-100, respectively. We repeat the experiments three times with different seeds for the random initialization of the network parameters and report averages and standard deviations.

Data splitting is important We first study the effect of data splitting by training each network on the whole training dataset (with no split). If our training framework had no benefit, then we would expect training on the full dataset to be beneficial, since each network simply sees more labeled data. However, we observe a degradation of the overall performance – which is more pronounced on CIFAR-100. Specifically, we observe a 38.21% drop (from 52.40% to 14.19%) in the case of a symmetric-90%-noise and a 4.66% drop (from 76.78% to 72.12%) in the case of an asymmetric-40%-noise. This is because the larger the noise level, the greater the effect of memorization. This tells us that data splitting is an important part in reducing memorization, even though each network sees less labeled data.

Class balancing is highly beneficial when noise is high Second, to highlight the effect of class-balancing coefficient normalization, we generate soft labels as in Equation (1) and Equation (2) but without normalizing the JSD. Somewhat surprisingly, this yields a slight performance *increase* in low-label-noise scenarios. However, when the noise ratio is large (symmetric-90%-noise, asymmetric-40%-noise), we see that it causes a large performance degradation: there is a drop of 15.71% (from 91.25% to 75.54%) for symmetric-90%-noise on CIFAR-10; and there are drops of 19.03% (from 52.40% to 33.37%) for symmetric-90%-noise and 5.19% (from 76.78% to 71.59%) for asymmetric-40%-noise on CIFAR-100. Moreover, when class-balancing normalization is not used, it can cause divergence in training. This yields a big performance gap between the best and the average (of the last 10 epochs) accuracy, especially in case of high noise scenarios (i.e in Table 5, Best: 71.59% vs. Last: 60.35% for asymmetric-40%-noise). This shows the importance of taking into account the class-wise difficulty, especially in the case of high noise ratio – as first demonstrated by (Karim et al., 2022).

Cross-split label correction is crucial Third, we demonstrate the benefit of cross-split label correction. When we only use the assigned label with no correction, we find a huge performance degradation – especially on CIFAR-100, which is known (Pleiss et al., 2020) to contain many

Table 5. Ablation study on CIFAR-100: Test accuracy (%) of different settings on CIFAR-100 with varying noise rates (50% - 90% for Symmetric and 10% - 40% for Asymmetric noise). With its higher difficulty than CIFAR-10, each component of *CrossSplit* is crucial when the noise ratios are high. Mean and standard deviation of best and average of last 10 epochs are calculated over 3 repetitions of the experiments. The best results are highlighted in **boldfaced** and scores that differ from them by more than 5% are marked in red.

Noise type	Symmetric				Asymmetric			
Noise ratio	50%		90%		10%		40%	
Method	Best	Last	Best	Last	Best	Last	Best	Last
CrossSplit	75.72±0.18	75.50±0.18	**52.40**±1.78	**52.05**±1.94	80.71±0.05	80.50±0.06	**76.78**±0.66	**76.56**±0.55
w/o data splitting	73.63±0.18	73.36±0.14	14.19±1.30	13.28±2.21	78.97±0.07	78.77±0.43	72.12±0.43	71.83±0.42
w/o class-balancing normalization	**77.67**±0.03	**77.17**±0.17	33.37±0.52	18.53±0.19	**82.86**±0.14	**82.57**±0.18	71.59±0.28	60.35±0.37
w/o cross-split label correction	70.20±0.16	65.74±0.10	31.77±0.32	15.93±0.21	82.38±0.16	82.10±0.23	69.61±0.65	59.67±0.11

Table 6. Performance (%) under extreme label noise on CIFAR-100. The table shows the best accuracy and the average accuracy over the last 10 epochs. (*) denotes the results we obtain by re-running their publicly available code.

Noise type	Symmetric					
Noise ratio	90%		92%		95%	
Method	Best	Last	Best	Last	Best	Last
UNICON	44.82	44.51	32.08*	31.85*	19.12*	18.14*
CrossSplit (ours)	**52.40**	**52.05**	**46.25**	**45.85**	**29.97**	**29.57**

more ambiguous examples compared to CIFAR-10. Especially, when the noise ratio is large (symmetric-90%-noise, asymmetric-40%-noise) on CIFAR-100, there are drops of 20.63% (from 52.40% to 31.77%) for symmetric-90%-noise and 7.17% (from 76.78% to 69.61%) for asymmetric-40%-noise. This demonstrates the value of our label correction procedure using the peer network.

In Appendix B.1, we additionally study the effect of the size of training set and the number of splits on performance.

5. Conclusion

This paper introduces a new framework for learning with noisy labels, which builds and improves upon existing methods based on label correction and co-teaching techniques. By using a pair of networks trained on two disjoint parts of the labeled dataset, our method bypasses the sample selection procedure used in recent state-of-the-art methods, which can be subject to selection errors. We propose data splitting, cross-split label correction by the peer network prediction, and class-balancing coefficient normalization that is effective in dealing with noisy labels. Our experimental results demonstrate the success of the method at mitigating the memorization of noisy labels, and show that it achieves state-of-the-art classification performance on several standard noisy benchmark datasets: CIFAR-10, CIFAR-100, and Tiny-ImageNet, with a variety of noise ratios. Most importantly, we also demonstrate that our method outperforms the state-of-the-art on the naturally noisy dataset mini-WebVision, which brings our model closer to real world application.

We restricted our study to image classification tasks in this paper. This is also the stage of many prior works in this line of research. However, we expect learning with noisy labels to come with its own challenges in other domains such as text classification (Zhu et al., 2022). Extending our study to this domain is an interesting avenue for future work.

Acknowledgements

This research was supported by Samsung and was enabled in part by compute resources provided by Mila (mila.quebec), Calcul Québec (calculquebec.ca), the Digital Research Alliance of Canada (alliancecan.ca), and by support from the Canada CIFAR AI Chair Program. Simon Lacoste-Julien is a CIFAR Associate Fellow in the Learning Machines & Brains program. We also thank the anonymous reviewers for their comments and suggestions.

References

Arazo, E., Ortego, D., Albert, P., O'Connor, N., and McGuinness, K. Unsupervised label noise modeling and loss correction. In *ICML*, pp. 312–321, 2019. 1, 4, 5, 6, 7, 8

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. In *ICML*, pp. 233–242, 2017. 1

Baldock, R., Maennel, H., and Neyshabur, B. Deep learning through the lens of example difficulty. In *NIPS*, volume 34, pp. 10876–10889, 2021. 1

Blum, A. and Mitchell, T. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100, 1998. 2, 5

Chen, P., Liao, B. B., Chen, G., and Zhang, S. Understand-

ing and utilizing deep neural networks trained with noisy labels. In *ICML*, pp. 1062–1070, 2019. 8

D'souza, D., Nussbaum, Z., Agarwal, C., and Hooker, S. A tale of two long tails. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2021. 1

Feldman, V. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020. 5

Goldberger, J. and Ben-Reuven, E. Training deep neural-networks using a noise adaptation layer. In *ICLR*, 2017. 4

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *NIPS*, 31, 2018. 1, 2, 4, 5, 8

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016a. 6

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 630–645. Springer, 2016b. 6

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, pp. 2304–2313, 2018. 8

Karim, N., Rizve, M. N., Rahnavard, N., Mian, A., and Shah, M. Unicon: Combating label noise through uniform selection and contrastive learning. In *CVPR*, pp. 9676–9686, 2022. 1, 3, 4, 5, 6, 7, 8, 12, 13, 15, 16

Kim, Y., Yun, J., Shon, H., and Kim, J. Joint negative and positive learning for noisy labels. In *CVPR*, pp. 9442–9451, 2021. 6, 7

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009. 5

Le, Y. and Yang, X. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 5

Li, J., Wong, Y., Zhao, Q., and Kankanhalli, M. S. Learning to learn from noisy labeled data. In *CVPR*, pp. 5051–5059, 2019. 1, 5

Li, J., Socher, R., and Hoi, S. C. H. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020. 1, 3, 4, 5, 6, 7, 8, 12

Li, S., Xia, X., Ge, S., and Liu, T. Selective-supervised contrastive learning with noisy labels. In *CVPR*, pp. 316–325, 2022. 6, 7

Li, W., Wang, L., Li, W., Agustsson, E., and Gool, L. V. Webvision database: Visual learning and understanding from web data. *CoRR*, 2017a. 1, 6

Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., and Li, L.-J. Learning from noisy labels with distillation. *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1928–1936, 2017b. 5

Liu, S., Niles-Weed, J., Razavian, N., and Fernandez-Granda, C. Early-learning regularization prevents memorization of noisy labels. In *NIPS*, volume 33, pp. 20331–20342, 2020. 1, 6, 7, 8

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 12

Lu, Y. and He, W. Selc: Self-ensemble label correction improves learning with noisy labels. In *IJCAI*, 2022. 1, 3, 4, 5, 6, 7, 8

Ma, F., Meng, D., Xie, Q., Li, Z., and Dong, X. Self-paced co-training. In *ICML*, pp. 2275–2284. PMLR, 2017. 5

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S., Xia, S., Wijewickrema, S., and Bailey, J. Dimensionality-driven learning with noisy labels. In *ICML*, pp. 3355–3364, 2018. 5

Ma, X., Huang, H., Wang, Y., Romano, S., Erfani, S., and Bailey, J. Normalized loss functions for deep learning with noisy labels. In *ICML*, pp. 6543–6553, 2020. 4

Malach, E. and Shalev-Shwartz, S. Decoupling "when to update" from "how to update". In *NIPS*, volume 30, 2017. 8

Ortego, D., Arazo, E., Albert, P., O'Connor, N. E., and McGuinness, K. Multi-objective interpolation training for robustness to label noise. In *CVPR*, pp. 6606–6615, 2021. 6, 7

Pleiss, G., Zhang, T., Elenberg, E., and Weinberger, K. Q. Identifying mislabeled data using the area under the margin ranking. In *NIPS*, volume 33, pp. 17044–17056, 2020. 1, 8

Reed, S. E., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. In *ICLR (Workshop)*, 2015. 1, 3, 4, 5, 6, 7

Sarfraz, F., Arani, E., and Zonooz, B. Noisy concurrent training for efficient learning under label noise. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3159–3168, 2021. 8

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C. A., Cubuk, E. D., Kurakin, A., and Li, C.-L. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NIPS*, volume 33, pp. 596–608, 2020. 2, 4

Song, H., Kim, M., and Lee, J.-G. Selfie: Refurbishing unclean samples for robust deep learning. In *ICML*, pp. 5907–5915, 2019. 1

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. Joint optimization framework for learning with noisy labels. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560, 2018a. 5

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. Joint optimization framework for learning with noisy labels. In *CVPR*, pp. 5552–5560, 2018b. 5

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. Learning from massive noisy labeled data for image classification. In *CVPR*, pp. 2691–2699, 2015. 1

Yu, X., Han, B., Yao, J., Niu, G., Tsang, I., and Sugiyama, M. How does disagreement help generalization against label corruption? In *ICML*, pp. 7164–7173, 2019. 1, 2, 4, 5, 8

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017. 1

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2, 4, 6, 7, 8

Zhang, Y., Zheng, S., Wu, P., Goswami, M., and Chen, C. Learning with feature-dependent label noise: A progressive approach. In *ICLR*, 2020. 4, 5

Zhang, Z. and Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NIPS*, volume 31, 2018. 4

Zhu, D., Hedderich, M. A., Zhai, F., Adelani, D. I., and Klakow, D. Is bert robust to label noise? a study on learning with noisy labels in text classification. In *ACL*, 2022. 9

A. Implementation Details

A.1. Detail on Relaxation Parameter

As mentioned in Equation (2) of the main paper, we use a relaxation parameter γ as a way to control the range of the combination coefficients in our definition of the soft labels. In our experiments, γ gradually increases from 0.6 to 1.0 during training according to the following schedule:

$$\gamma = \begin{cases} 0.6, & \text{if } epoch \in [E_{\text{warm}}, E_{\text{warm}} + 2\delta] \\ 0.8, & \text{else if } epoch \in [E_{\text{warm}} + 2\delta, E_{\text{warm}} + 3\delta] \\ 1.0, & \text{otherwise} \end{cases} \tag{A.1}$$

where the parameter δ determines the relaxation period. We empirically set δ to 10.

It is interesting to show how the value of γ affects the results. We investigate two additional settings: (i) constant small (= 0.6) γ, (ii) constant large (= 1.0) γ. The results are shown in Table A.1.

Table A.1. Effect of γ setting. The best scores are **boldfaced**, and the second best ones are underlined.

Noise type	Symmetric			
Dataset	CIFAR-10		CIFAR-100	
Noise ratio	50%	90%	50%	90%
0.6 (constant)	**96.38**	89.65	**79.16**	45.77
1.0 (constant)	96.29	**93.56**	75.24	<u>50.71</u>
$0.6 \rightarrow 0.8 \rightarrow 1.0$ (ours)	<u>96.34</u>	<u>91.25</u>	<u>75.72</u>	**52.40**

We find different outcomes depending on the noise ratio. For a 50% noise ratio, setting (i) performs better than (ii); for a 90% noise ratio this is the other way around. The setting of our paper yields a good performance across various noise ratios.

A.2. Training Details

Table A.2. Training details on CIFAR-10, CIFAR-100, Tiny-ImageNet and mini-WebVision datasets.

Dataset	CIFAR-10	CIFAR-100	Tiny-ImageNet	mini-WebVision
Batch size	256	256	40	128
Network	PRN-18	PRN-18	PRN-18	ResNet-18
Epochs	300	300	360	140
Optimizer	SGD	SGD	SGD	SGD
Momentum	0.9	0.9	0.9	0.9
Weight decay	5e-4	5e-4	5e-4	5e-4
Initial LR	0.1	0.1	0.005	0.02
LR scheduler	Cosine Annealing LR			Multi-Step LR
T_{max}/LR decay factor	300	300	360	0.1 (80, 105)
Warm-up period	10	30	10	1

The training details are summarized in Table A.2. For CIFAR-10 and CIFAR-100, we train each network using stochastic gradient descent (SGD) optimizer with momentum 0.9 and a weight decay of 0.0005. Training is done for 300 epochs with a batch size of 256. We set the initial learning rate as 0.1 and use a a cosine annealing decay (Loshchilov & Hutter, 2017). Just like in (Li et al., 2020; Karim et al., 2022), a warm-up training on the entire dataset is performed for 10 and 30 epochs for CIFAR-10 and CIFAR-100, respectively.

For Tiny-ImageNet, we use SGD with momentum 0.9, a weight decay of 0.0005, and a batch size of 40. We train each network for 360 epochs, which includes a warm-up training of 10 epochs.

For mini-WebVision, we use SGD with momentum 0.9, a weight decay of 0.0005, and a batch size of 128. We train the networks for 140 epochs with a warm-up period. We also set the initial learning rate to 0.02 and decay it with decay factor

0.1 with intervals of 80 and 105.

B. Additional Studies

B.1. Effects of Training Set Size on Performance

There might be a tradeoff between a potential performance loss due to halving of the dataset for the two individual networks (although it is mitigated by our use of the semi-supervised setup), and the performance loss due to noise memorization. Our empirical results show that this tradeoff is worthwhile for learning with noisy labels (see Table 4 and Table 5). In Section 4.6, our ablation study shows that when we do not perform data splitting (i.e. use the whole dataset), the results are consistently worse.

In addition, we investigate the effect of training set size on performance further. One way to do this is to extend the method to multiple networks trained on multiple splits. Depending on the number of data splits, the ratios of labeled and unlabeled datasets (S_{labeled}, $S_{\text{unlabeled}}$) can vary. We set the number of data splits from two (default) to four. Figure B.1 shows five possible scenarios, (a–e). When we train two (a), three (b–c) or four (d–e) models, each model is trained via the semi-supervised training process with the data configuration depicted on the left side of the model. Then, each model performs cross-split label correction using the peer network prediction (the remaining models' predictions) for the labeled data.

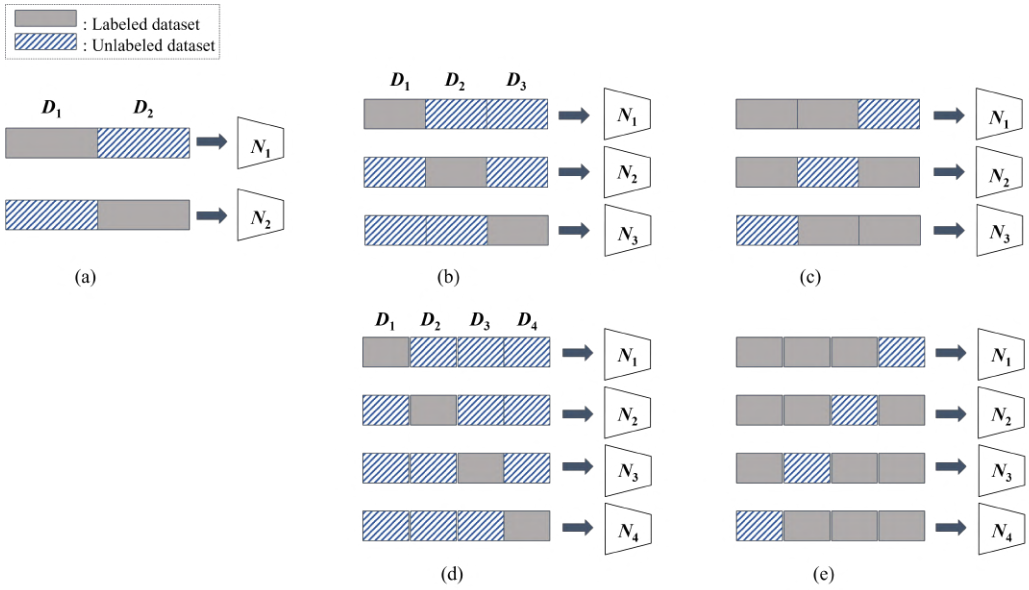

Figure B.1. Visualizations of five possible data configurations: (a) (N_{splits}, S_{labeled}, $S_{\text{unlabeled}}$) = (2, 0.50, 0.50), (b) (3, 0.33, 0.67), (c) (3, 0.67, 0.33), (d) (4, 0.25, 0.75), and (e) (4, 0.75, 0.25)

Table B.1 demonstrates that there is a considerable difference in performance due to the ratio of labeled information especially in the case of 90% noise on CIFAR-100. That is, the larger the ratio of labeled information, the better the performance ((d) $S_{\text{labeled}} = 0.25 <$ (b) $0.33 <$ (a) $0.50 <$ (c) $0.67 <$ (e) 0.75). However, when the amount of noise is small, the performance difference due to the proportion of labeled datasets is not large and (a) ($S_{\text{labeled}} = 0.50$) performs better than other cases in most cases. This results in an optimal number of splits that is 2 in favour of our original setting.

B.2. Effect of Contrastive Loss

As mentioned in Equation (5), following (Karim et al., 2022), we use a contrastive loss $\mathcal{L}_{\text{contrastive}}$ in addition to the semi-supervised loss for the training of the two networks. Here we show ablation over this unsupervised learning component. The results are shown in Table B.2. We observe that the contrastive loss is particularly helpful in improving the performance in a high noise regime (90%).

Table B.1. Effect of the number of splits and data configurations: Test accuracy (%) of different setting on CIFAR-10 with varying label noise ratios (50% and 90% for symmetric noise). We see that there is a minor difference when changing the number of splits and data configurations with lower noise ratios, but a large degradation in performance if it is changed for high noise ratios. Among five scenarios, the setting of our paper (a) yields a good performance across various noise ratios. Mean and standard deviation of best and average of last 10 epochs are calculated over 3 repetitions of the experiments. The best results are highlighted in **boldfaced**.

Noise type	Symmetric							
Dataset	CIFAR-10				CIFAR-100			
Noise ratio	50%		90%		50%		90%	
(N_{splits}, $S_{labeled}$, $S_{unlabeled}$)	Best	Last	Best	Last	Best	Last	Best	Last
(a) (2, 0.50, 0.50)	**96.34**±0.05	**96.23**±0.07	**91.25**±0.79	**91.02**±0.77	**75.72**±0.18	**75.50**±0.18	52.40±1.78	52.05±1.94
(b) (3, 0.33, 0.67)	96.10±0.15	95.95±0.09	88.73±0.72	88.53±0.75	75.46±0.06	75.27±0.02	50.73±0.65	50.15±0.84
(c) (3, 0.67, 0.33)	96.02±0.06	95.95±0.06	88.74±0.05	88.57±0.07	74.97±0.10	74.79±0.10	52.46±1.98	52.27±1.92
(d) (4, 0.25, 0.75)	**95.59**±0.19	**95.50**±0.16	88.13±0.25	87.89±0.36	73.83±0.30	73.66±0.28	46.04±0.57	44.76±0.77
(e) (4, 0.75, 0.25)	96.08±0.05	95.97±0.02	88.74±1.00	88.57±0.96	75.71±0.14	75.45±0.25	**53.79**±0.94	**53.61**±0.97

Table B.2. Effect of contrastive loss on CIFAR-100 with varying label noise ratios (50% and 90% for symmetric noise). The best scores are **boldfaced**.

Noise type Noise ratio	Symmetric 50%		 90%	
Method	Best	Last	Best	Last
CrossSplit	**75.72**	75.50	**52.40**	**52.05**
CrossSplit w/o $L_{contrastive}$	75.68	**75.58**	31.42	31.15

B.3. Effect of Warm-up Epochs

We perform ablation studies where we vary the number of warm-up epochs. We investigate two additional settings on CIFAR-10 and CIFAR-100: (i) warm-up = (20 epochs on CIFAR-10, 45 epochs on CIFAR-100), (ii) warm-up = (30, 60). We find that the number of warm-up epochs makes very little difference, with a maximum difference of only 0.17 (CIFAR-10) and 0.27 (CIFAR-100), so it appears that the specific choice for warm-up makes very little actual difference.

Table B.3. Test accuracy (%) for different E_{warm} setting with symmetric 50% label noise. It appears that the specific choice for warmup makes very little actual difference.

Noise type/Noise ratio	Symmetric 50%	
Dataset	CIFAR-10	CIFAR-100
(10, 30) (ours)	96.34	75.72
(20, 45)	96.24	75.45
(30, 60)	96.17	75.46

B.4. Running Time for Training

We provide running time comparisons between *CrossSplit* and UNICON in the table below. We observe that UNICON has a lower running time on average (281 seconds / epoch versus 369 seconds / epoch for *CrossSplit* in the case of a 20% noise ratio), only partially compensated by a slower convergence (350 epochs versus 300 for *CrossSplit*).

We also note that UNICON running time is more dependent on the noise ratio. This is to be expected from dynamical sample selection methods, for which the size of the labeled dataset the networks are trained on reduces as the noise ratio increases. The following results are on CIFAR-10, reporting seconds / epoch (average of the next 5 epochs after warm-up), run on one RTX8000 GPU. UNICON uses 350 epochs while *CrossSplit* uses 300 epochs.

Table B.4. Running time comparison on CIFAR-10 with varying noise rates (20% - 90% for symmetric noise).

Noise type	Symmetric				
Noise ratio	20%	50%	80%	90%	Total Time
UNICON	281.62s	254.08s	246.91s	227.30s	252.48s * 350 = 24.5h
CrossSplit (ours)	369.47s	370.78s	370.19s	373.01s	370.86s * 300 = 30.9h

B.5. T-SNE Visualization

In this section, we provide a visual comparison of the features (penultimate layer) learned by UNICON (Karim et al., 2022) and *CrossSplit*. Both are trained with PreAct ResNet-18. Figure B.2 and Figure B.3 show the class distribution of the features corresponding to test images on CIFAR-10 and CIFAR-100 with asymmetric noise (40%) and symmetric noise (50%, 90%), respectively. This suggests that the representations learned by *CrossSplit* do a better job at separating the classes than UNICON.

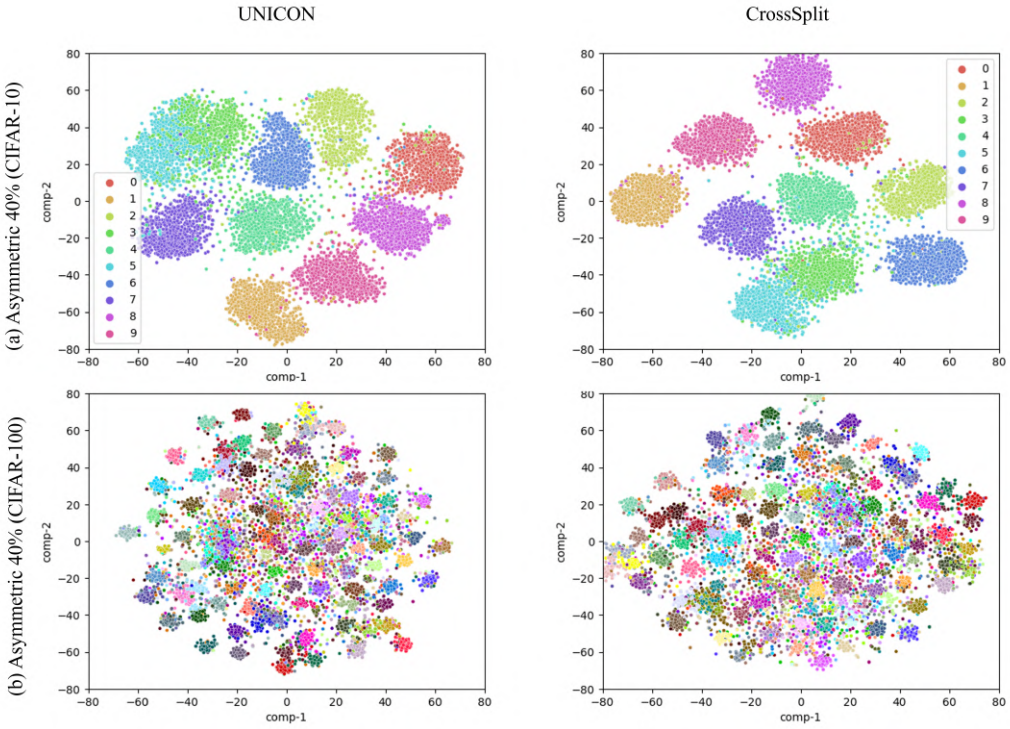

Figure B.2. T-SNE visualizations of learned features of test images by UNICON (Karim et al., 2022) and CrossSplit with asymmetric noise of 40%. In general, the clusters for CrossSplit are significantly better separated than for UNICON.

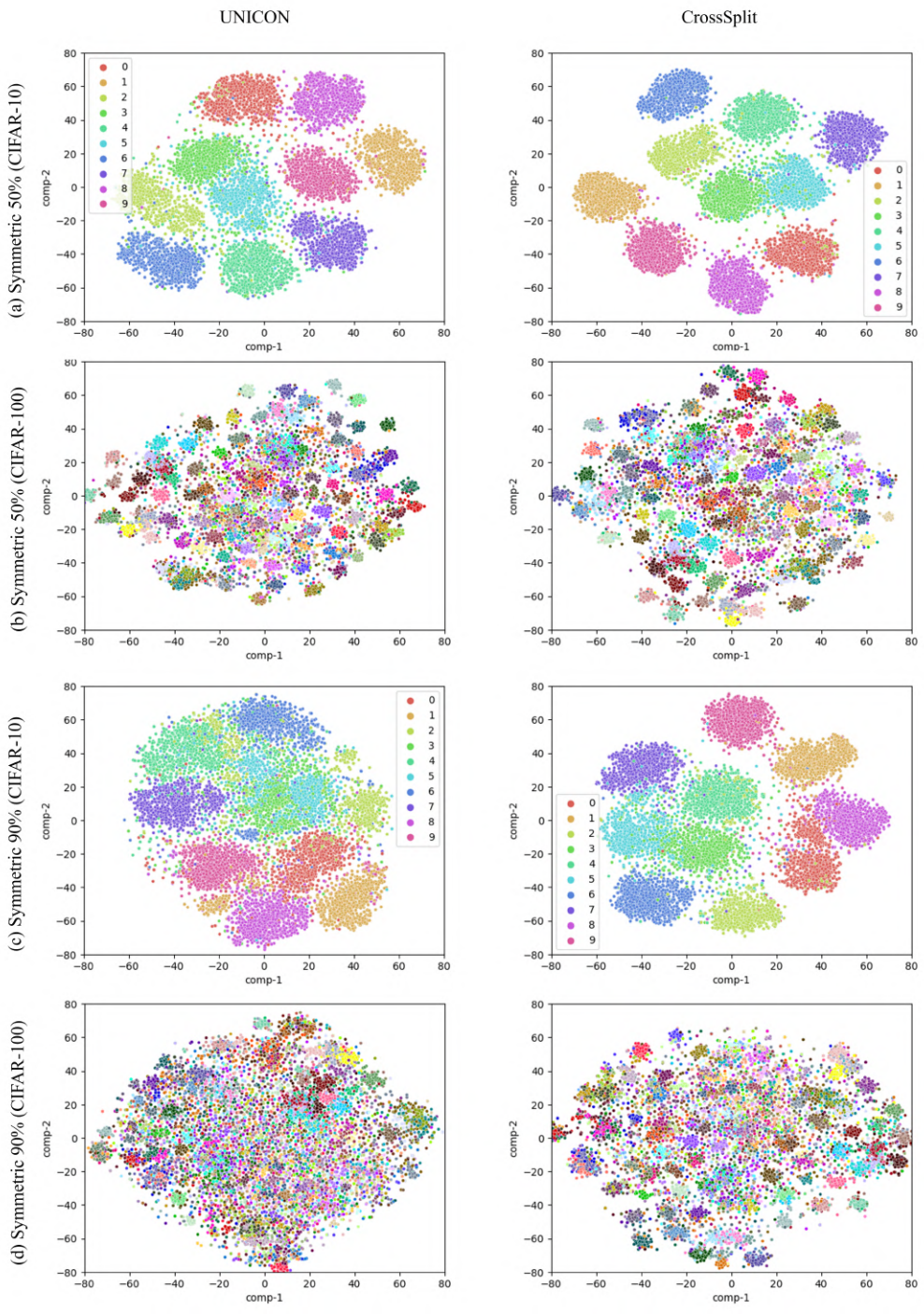

Figure B.3. T-SNE visualizations of learned features of test images by UNICON (Karim et al., 2022) and CrossSplit with symmetric noise of 50% and 90%. In general, the clusters for CrossSplit are significantly better separated than for UNICON. This is evidence for the superior representation learned by reducing memorization of noisy labels through CrossSplit.