Co-learning: Learning from Noisy Labels with Self-supervision

Cheng Tan¹2, Jun Xia¹2, Lirong Wu¹2, Stan Z. Li¹2 ∗
¹AI Lab, School of Engineering, Westlake University.
²Institute of Advanced Technology, Westlake Institute for Advanced Study
18 Shilongshan Road, Hangzhou 310024, Zhejiang Province, China
{tancheng, xiajun, wulirong, stan.zq.li}@westlake.edu.cn

ABSTRACT
Noisy labels, resulting from mistakes in manual labeling or webly data collecting for supervised learning, can cause neural networks to overfit the misleading information and degrade the generalization performance. Self-supervised learning works in the absence of labels and thus eliminates the negative impact of noisy labels. Motivated by co-training with both supervised learning view and self-supervised learning view, we propose a simple yet effective method called Co-learning for learning with noisy labels. Co-learning performs supervised learning and self-supervised learning in a cooperative way. The constraints of intrinsic similarity with the self-supervised module and the structural similarity with the noisily-supervised module are imposed on a shared common feature encoder to regularize the network to maximize the agreement between the two constraints. Co-learning is compared with peer methods on corrupted data from benchmark datasets fairly, and extensive results are provided which demonstrate that Co-learning is superior to many state-of-the-art approaches. The code is available at GitHub.

CCS CONCEPTS
• Computing methodologies → Computer vision.

KEYWORDS
label noise, self-supervised learning, computer vision

ACM Reference Format:
Cheng Tan¹2, Jun Xia¹2, Lirong Wu¹2, Stan Z. Li¹2 . 2021. Co-learning: Learning from Noisy Labels with Self-supervision. In Proceedings of the 29th ACM International Conference on Multimedia (MM ’21), October 20–24, 2021, Virtual Event, China. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3474085.3475622

1 INTRODUCTION
The success of deep learning relies on carefully labeled data in large-scale. However, precise annotations are extremely expensive and time-consuming. To alleviate the problem, inexpensive alternatives are often used, such as web crawling [35] or completing annotations with crowd-sourcing and online queries. Unfortunately, these alternative methods can often lead to noisy labels. However, deep neural networks can easily overfit to noisy labels, as shown in recent research [18, 22, 34, 40], and this can dramatically degrade the generalization performance.

Several studies have been conducted to investigate learning with noisy labels, where semi-supervised learning frameworks play crucial roles [1, 8, 17, 25, 38]. Most of these deploy unsupervised learning to obtain information about label-independent features and then do further training using noisy labels in a supervised learning manner. These methods commonly leverage labels through two views: (1) supervised learning taking advantage of direct supervision with labels and (2) unsupervised learning exploiting intrinsic features from data to combat label noise. The core challenge here is how to combine the two views effectively. Araoz et al. [1] design a beta mixture model as an unsupervised generative model of sample loss values and then adopt MixUp [41] augmentation to assure reliable convergence under extreme noise levels. Li et al. [17] models sample loss with Gaussian mixtures to divide training data into unlabeled data and labeled data, and then apply the state-of-the-art
semi-supervised learning method MixMatch [2] to deal with the
divided sets. However, the above methods are not end-to-end and
the step-by-step learning manner increases complexity in both time
and space.

Co-training and model ensembling have been shown to be ben-
eficial in learning with noisy labels [11, 23, 37, 39]. Decoupling
[23] trains two networks simultaneously and updates them using
instances with different predictions. Co-teaching [11] also trains
two networks simultaneously and selects small-loss data to teach
the peer network during the training process. Co-teaching+ [39]
follows a similar scheme as Co-teaching but with a scheme which
selects small-loss data from disagreement data. JoCoR [37] main-
tains two networks but trains them as a whole with a joint loss
to make predictions of them get closer. These methods are based
on the assumption that two networks can provide two different
views of the data, but the extra information gain they bring is very
limited since the differences between two networks of the same
architecture mainly come from random initialization. A compari-
son between Co-learning and other co-training-like methods is
illustrated in Figure 1.

Motivated by semi-supervised learning and co-training, we pro-
pose a simple yet effective learning paradigm called “Co-learning”
to combat the problem with noisy labels, in which self-supervised
learning is introduced as a feature-dependent view to assist su-
ervised learning. Different from those co-training-like methods,
Co-learning has a shared feature encoder with two exclusive heads.
While the projection head performs a self-supervised learning and
exploits feature-dependent information through the intrinsic sim-
ilarity, the classifier head performs a vanilla supervised learning
and learns from label-dependent information. In addition, a struc-
tural similarity loss is utilized between the two exclusive heads to
regularize the classifier and avoid bias caused by noisy labels. This
is based on the constraint that output from the projection head and
that from the classifier head should share similar structure pairwise.
Unlike the other methods, Co-learning can be implemented conve-
niently without the need for such prior knowledge as noise rates,
data distributions, and additional clean samples, thus avoiding the
needed hyperparameters thereof.

We conducted experiments on simulated and real-world noisy
datasets, including CIFAR-10, CIFAR-100, Animal-10N [32], and
Food-101N [16] datasets. Among these datasets, Animal-10N is a
noisy dataset that contains confusing images for manual annota-
tions, and Food-101N is a webly dataset that directly collects data
from the Web. Comparative results demonstrate that Co-learning is
superior to state-of-the-art methods and robust to high-level noise
in labels.

2 RELATED WORK

2.1 Learning with noisy labels

Various methods have been studied towards robust learning with
noisy labels, and we briefly review existing works on learning with
noisy labels in this section.

Noise transition matrix. Some of the existing methods estimate
the label transition matrix. Forward-correction [29] performs for-
ward correction by multiplying the estimated transition matrix with
the softmax outputs during the forward propagation step. Gold loss
correction [14] leverages available trusted labels for loss correction
and utilizes trusted labels’ confusion matrices to obtain an accu-
rate transition matrix. However, estimating the transition matrix is
challenging in real-world settings and maybe infeasible for finding
closed-form solutions.

Sample selection. Another direction of handling noisy labels with-
out noise transition matrix is training on selected samples. These
approaches try to select “clean” instances according to certain rules.
In particular, MentorNet [15] pretrains a teacher network for se-
lecting clean instances to guide the training of a student network.
Co-teaching [11] chooses clean instances from a network to feed its
peer network for updating parameters. Co-teaching+ [39] and Jo-
CoR [37] also highlight the small-loss sample selection to explicitly
assist the co-training procedures of two networks. INCV [5] divides
noisy training data randomly to apply cross-validation and removes
large-loss samples. This family of methods avoids the risk of error
correction by simply excluding unreliable samples. Although these
methods have achieved significant improvements, they neglect the
potential supervisory signal of the large-loss training samples. Be-
side, either the noise rate or a clean validation set is required as
prior knowledge.

Semi-supervised learning. Several recent studies transform the
problem of learning with noisy labels into a semi-supervised learn-
ing task. There are mainly two directions in these methods: (1)
divide noisy training data into two sets where reliable samples are
treated as labeled data, and unreliable samples are treated as una-
leveled data, then perform certain existing semi-supervised learning
methods [1, 8, 17, 25, 38], (2) perform unsupervised learning on
noisy training data to assist the supervised learning [12, 24]. Our
method follows the second direction that utilizes unsupervised
learning to implicitly help to learn from noisy labels.

2.2 Self-supervised learning

Self-supervised learning is a subset of unsupervised learning that
designs pretext tasks to produce labels derived from the data it-
self. While solving pretext tasks, the model learns valuable rep-
resentations; that is, the feature encoder of the model is able to
extract useful features. There are various pretext tasks that can
help the model learn representations, such as colorization [42],
inpainting [28], rotation [9], jigsaw [26] and contrastive learning
[4, 6, 7, 10, 13]. Among them, recent studies on contrastive learning
have become a dominant component in self-supervised learning
methods. Contrastive learning aims at grouping similar samples
closer and diverse samples away from each other. A typical con-
trastive learning pipeline is shown in Figure 2. To measure the
similarity between the feature representations, contrastive learning
utilizes a metric that we call intrinsic similarity in this paper.

Current state-of-the-art contrastive learning methods mainly
rely on a combination of intrinsic similarity and a set of image
transformations. MoCo [13] builds a dynamic dictionary with a
queue and a momentum encoder to obtain large and consistent
dictionaries of visual representations. SimCLR [6] shows that the
composition of data augmentations is crucial and a nonlinear projection head will further improve the quality of the learned representations. While contrastive learning may fall into collapsed solutions like outputting the same representations for all samples, BYOL [10] takes advantage of both the target network and the slow-moving average online network to avoid collapsing. To simplify the framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation. SwAV [4] introduces an online clustering framework in BYOL, SimSiam [7] proposes Siamese networks with a stop-gradient operation.
which perform contrastive prediction and supervised classification

where fitting to noisy labels. Then we create new training samples by

Taking $f$ as the feature encoder, the cross-entropy loss is a common supervised loss to measure the

pretext tasks. This indicates the same data has structural similarity through different

transformations of $x(i)$ is $\hat{x}(i) = h(x(i); \theta_3)$. The intrinsic similarity loss is calculated between $u_2(i)$ and $v_3(i)$ by taking $(u_2(i), v_3(i))$ as a positive pair and $\{(u_2(i), v_2(i)), (v_3(i), v_3(i))\}_{i \neq i}$ as the negative pairs. This indicates the same data has intrinsic similarity through different transformations. The two different heads which perform contrastive prediction and supervised classification simultaneously can be regarded as two different pretext tasks. This indicates the same data has structural similarity through different pretext tasks.

Specifically, the complete loss function $L$ is a linear combination of the supervised loss $L_{sup}$, intrinsic similarity loss $L_{int}$, and structural similarity loss $L_{str}$ as follows:

\[
L = L_{sup} + L_{int} + L_{str}
\]

Supervised classifier loss. For multi-class classification problems, the cross-entropy loss is a common supervised loss to measure the relative entropy between the corrupted label $\hat{y}$ and its corresponding prediction $\hat{y}$.

\[
L_{ce}(\hat{y}, \hat{y}) = -\sum_{i=1}^{N} \hat{y}_i \log(\hat{y}_i)
\]

Considering contrastive learning requires a long training period comparing to its supervised counterpart. Co-learning deploys MixUp [41] augmentation to slow down the convergence of the feature encoder $f$ and avoid the supervised learning quickly overfitting to noisy labels. Then we create new training samples by linearly interpolating a sample with another sample (indexed by $m(i)$) randomly chosen from the same mini-batch:

\[
\hat{x}(i) = \lambda x(i) + (1 - \lambda) x(m(i))
\]

$\hat{y}(i) = \lambda \hat{y}(i) + (1 - \lambda) \hat{y}(m(i))$

where $\lambda \sim \text{Beta}(a, a)$.

Then, the MixUp version of the supervised loss is formulated as:

\[
L_{sup} = -\sum_{i=1}^{N} \left[ \lambda \hat{y}_i \log(\hat{y}(i)) + (1 - \lambda) \hat{y}_i \log(\hat{y}(m(i))) \right]
\]

Intrinsic similarity loss. In the contrastive prediction task, projections of the same image from different views should be similar. The intrinsic similarity between $v_2(i)$ and $v_3(i)$ (the projections of the two strongly transformed versions $\hat{x}(i)$ and $\hat{x}(i)$ from $x(i)$) is measured by the contrastive loss function as in InfoNCE [27]:

\[
t(v_a(i), v_b(i)) = -\log \frac{\exp(D(v_a(i), v_b(i))/\tau)}{\sum_{j=1, \neq j} \sum_{t, t \in \{2, 3\}} \exp(D(v_a(i), v_b(i))/\tau)}
\]

where $D(v_a, v_b) = \frac{v_a(v_b)}{||v_a|| ||v_b||}$ is the pairwise similarity between $v_a$ and $v_b$, and $\tau$ is a temperature parameter. The intrinsic similarity loss is thus defined as:

\[
L_{int} = \sum_{i=1}^{N} t(v_2(i), v_3(i)) + t(v_2(i), v_2(i))
\]

Structural similarity loss. The classifier $g$ and the projection head $h$ share the same encoder network, so they jointly influence the process of the encoder learning. However, the classifier could be misled by noisy labels. To overcome this problem, we impose a structure-preserving constraint [19, 20] on pairs of features from $g(\cdot)$ and the corresponding ones from $h(\cdot)$, as illustrated in Figure 4. The structure similarity loss is defined as follows.

Figure 4: The corresponding outputs from $g$ and $h$ should have similar relational structures between pairs of points. Left: transformation $\hat{y} \rightarrow \hat{v}$; Right: The relationship between transformed points is similar to the previous.

First, we convert the Euclidean distances $d(\hat{v}(i), \hat{v}(j)), d(\hat{y}(i), \hat{y}(j))$ into the similarity metrics $p(d(\hat{v}(i), \hat{v}(j))), p(d(\hat{y}(i), \hat{y}(j))) \in [0, 1]$ that satisfy $\lim_{d \to +\infty} p(d) = 0$ and $\lim_{d \to 0} p(d) = 1$. The similarity metric is formulated as:

\[
p(d) = C_\sigma \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} \left( \frac{d^2}{2\sigma^2} \right)}
\]

where $C_\sigma = \sigma \sqrt{2\pi}$ is a constant normalizing the similarity metric in $[0, 1]$, and we set $\mu = 0$, $\sigma = 0.5$ empirically.

Then the structural similarity loss is defined as the KL-divergence between similarity metrics converted from the projection $z$ and logits $\hat{y}$.
We consider two types of synthetic label noise: symmetric noise and asymmetric noise. We corrupt the original labels manually. Similar to previous works [29, 30, 33], we use the following two types of synthetic label noise:

- **Symmetric noise**: We corrupt each class to the next class with probability $p$. For example, for the CIFAR-10 dataset, we corrupt the class "dog" with probability $p$ to become the class "cat". We generate labels by flipping each class to a similar class with probability $p$. This method is used in the symmetric noise setup. In the asymmetric noise setup, labels are generated by flipping each class to a similar class with probability $p$.

- **Asymmetric noise**: We generate labels by flipping each class to a different class with probability $p$. For example, for the CIFAR-10 dataset, we corrupt the class "dog" with probability $p$ to become the class "plane". We generate labels by flipping each class to a different class with probability $p$. This method is used in the asymmetric noise setup. In the asymmetric noise setup, labels are generated by flipping each class to a different class with probability $p$.

The proposed approach is delineated in Algorithm 1.

### Algorithm 1: Co-learning

**Require**: Encoder $f$ with parameter $\theta_1$, classifier $g$ with parameter $\theta_2$, projection head $h$ with parameter $\theta_3$, learning rate $\eta$, batch size $B$, epoch $T_{\text{max}}$

1. for $t = 1, 2, \ldots, T_{\text{max}}$
2. 
3. for $n = 1, 2, \ldots, \lfloor \frac{|D|}{B} \rfloor$
4. 
5. for $i = 1, 2, \ldots, |B|$
6. 
7. 
8. 
9. 
10. 
11. 
12. 
13. 
14. 
15. 
16. return $\theta_1$, $\theta_2$, $\theta_3$

### 4 EXPERIMENTS

#### 4.1 Experimental settings

**Datasets**: Experiments are conducted on four benchmark datasets: CIFAR-10, CIFAR-100, Animal-10N [32], and Food-101N [16] which are popularly used for evaluating models in learning from noisy labels [1, 31].

Since CIFAR-10 and CIFAR-100 are clean datasets with true labels, we need to corrupt the original labels manually. Similar to previous works [29, 30, 33], we consider two types of synthetic label noise: symmetric noise and asymmetric noise. Following [36, 44], label noise is uniformly distributed among all categories with probability $\frac{p}{N}$, $p \in [0, 1]$ in the symmetric noise setup. In the asymmetric noise setup, labels are generated by flipping each class to a similar class or the next class circularly with probability $p$. We corrupt CIFAR-10 and CIFAR-100 datasets by the label transition matrix $Q$, where $Q_{ij} = \Pr(\hat{y} = j | y = i)$ given that noisy $\hat{y}$ is flipped from clean $y$. Figure 6 shows an example of noise transition matrix on CIFAR-100.

Following previous works [37, 43], we evaluate our method on various noise rates $\eta$, with $\eta \in \{20\%, 50\%, 80\%\}$ for symmetric label noise and $\eta \in \{20\%, 30\%, 40\%\}$ for asymmetric label noise. We use no more than 40% because certain classes become theoretically indistinguishable for asymmetric noise larger than 50%. As for the asymmetric noise of CIFAR-10, following [29] the noisy labels were generated by mapping "truck" → "automobile", "bird" → "airplane", "deer" → "horse" and "cat" → "dog" with probability $p$. But for the asymmetric noise of CIFAR-100, we generate labels by flipping each class to the next class with probability $p$.

Besides, Animal-10N [32] is a real-world noisy dataset of human-labeled online images for ten confusing animals, with 50,000 training and 5,000 testing images. As people may label images with confusing patterns, this dataset has natural noisy data. Its noise rate is estimated at 8%. Food-101N is a webly noisy dataset that [16] collects 310k images from Google, Bing, Yelp, and TripAdvisor using the Food-101 [3] taxonomy. We train models on Food-101N and evaluate them on the test set of Food-101.

**Baselines**: We compare our Co-learning with the following state-of-the-art methods, and we implement all methods with their default settings by PyTorch based on their open-source code.

1. Standard cross-entropy, which refers to train on noisy datasets directly with standard cross-entropy loss. This method is applied as a simple baseline.
2. Decoupling [23], which updates the parameters only using samples that have different predictions from two classifiers.
3. Co-teaching [11], which trains two networks simultaneously and cross-updates parameters of their peer networks.
4. Co-teaching+ [39], which trains two networks simultaneously and cross-updates their peer networks with the samples with different predictions from two classifiers.
5. JoCoR [37], which trains two networks jointly and makes predictions from two classifiers similar by a Jensen-Shannon divergence.
6. APL [21], which proposes a framework that combines two robust loss functions that mutually boost each other.

**Network Structure and Optimizer**: We utilize ResNet-18 for CIFAR-10 and CIFAR-100 datasets. For Animal-10N and Food-101N datasets, we utilize ResNet-34.

For experiments on CIFAR-10 and CIFAR-100, Adam optimizer is applied with an initial learning rate of 0.001 and a momentum of 0.9. The batch size is fixed as 128. For each trial, we run 200 epochs, and the learning rate is linearly decayed to zero from 80 to 200 epochs. All the experiments are repeated five times with different random seeds, and we report the average test accuracy of five trials for each experiment. For experiments on the real-world dataset Animal-10N and Food-101N, Adam optimizer is also applied with an initial learning rate of 0.001 and a momentum of 0.9. The batch size is set to 64. We run each trial for 100 epochs.

**Measurement**: To evaluate the performance, the test accuracy is used, i.e., test accuracy = (# of correct predictions) / (# of the test). As the experiments of the synthetic noisy datasets are repeated five times, the MEAN and STD of the test accuracy are calculated. We have been highlighted the error bar for STD as a shade in each figure.

#### 4.2 Comparison with the State-of-the-Arts

**Results on CIFAR-10**: Figure 5 shows the test accuracy on CIFAR-10. Co-learning performs best in all six cases. In these figures, test accuracy of the standard cross-entropy first reaches a very high level and then decreases gradually, which indicates the memorization effects of neural networks. The higher the noise rates, the more serious the problems become. A robust training method toward learning with noisy labels should avoid overfitting on noisy training data.
The test accuracy of different algorithms is compared and reported in Table 2. In the easiest Symmetry-20% case, all approaches work well except standard cross-entropy, demonstrating their robustness to low-level label noise. When it comes to higher noisy rates like Symmetry-50% and Asymmetry-40%, Decoupling starts to overfit to noisy data as standard cross-entropy does. In the hardest Symmetry-80% case, Co-learning significantly outperforms other approaches and suffers less from noisy training data. Co-learning performs better than others in high noisy rates due to the feature-dependent information that comes from self-supervision.

**Table 2: Average test accuracy (%) on CIFAR-10 over the last 10 epochs.**

| Flipping-Rate | Standard      | Decoupling    | Co-teaching   | Co-teaching+ | JoCoR         | APL          | Co-learning |
|---------------|---------------|---------------|---------------|--------------|---------------|--------------|-------------|
| Symmetric-20% | 84.81±0.24    | 85.75±0.31    | 90.29±0.19    | 88.63±0.32   | 90.43±0.25    | 88.54±0.45   | 92.21±0.31  |
| Symmetric-50% | 61.49±0.58    | 61.93±0.82    | 63.43±3.89    | 76.27±2.80   | 66.00±0.53    | 76.51±1.73   | 84.49±0.34  |
| Symmetric-80% | 28.98±0.26    | 27.23±0.84    | 28.03±1.67    | 30.37±1.69   | 29.19±1.64    | 24.75±2.87   | 61.20±2.29  |
| Asymmetric-20%| 87.00±0.20    | 87.66±0.29    | 89.38±0.33    | 89.00±0.18   | 89.20±0.26    | 88.02±0.29   | 91.07±0.32  |
| Asymmetric-30%| 81.99±0.31    | 81.83±0.26    | 86.58±1.32    | 86.22±0.26   | 86.41±0.45    | 86.03±0.21   | 86.89±0.87  |
| Asymmetric-40%| 76.30±0.34    | 74.97±0.38    | 74.25±0.38    | 81.25±0.75   | 73.95±1.00    | 80.97±0.19   | 81.42±0.52  |

**Results on CIFAR-100.** Table 3 shows results on CIFAR-100. As there are only 10 classes in CIFAR-10 but 100 classes in CIFAR-100, the overall accuracy is much lower than the previous. The memorization effects of neural networks are severe in standard cross-entropy as well as other approaches like Decoupling and Co-teaching+. As we can see, the observations are consistently the same as those for the CIFAR-10 dataset, which is, Co-learning still works much better than other approaches. In the easiest Symmetry-20% case, only standard cross-entropy and Decoupling fail to combat noisy training data while others perform well. However, Co-learning works significantly better than other methods under high noisy rate cases such as Symmetry-50% and Symmetry-80%.

Figure 7 shows the test accuracy during the training process with different noise types. Co-learning performs more stable learning curves with much less overfitting than other methods.

**Results on Animal-10N.** Table 4 summarizes the test results on the real-world noisy labels using the Animal-10N dataset, where best denotes the optimal validation accuracy, and last denotes the final test accuracy. As Co-teaching, Co-teaching+, and JoCoR require a noisy rate estimation to assist their training, we roughly set it as 10%. Despite training with less prior knowledge, Co-learning consistently outperforms other methods.

**Results on Food-101N.** Table 5 illustrates the test results on Food-101N dataset, where best denotes the optimal validation accuracy,
Figure 7: Results on CIFAR-100 dataset.

| Flipping-Rate | Standard CE | Decoupling | Co-teaching | Co-teaching+ | JoCoR | APL | Co-learning |
|---------------|-------------|------------|-------------|--------------|-------|-----|-------------|
| Symmetric-20% | 57.79±0.44  | 56.18±0.32 | 64.28±0.32  | 55.40±0.71   | 62.29±0.71 | 59.21±0.50 | 66.58±0.15  |
| Symmetric-50% | 33.75±0.46  | 31.58±0.54 | 32.62±0.51  | 26.49±0.45   | 30.19±0.60 | 43.53±1.84 | 54.54±0.43  |
| Symmetric-80% | 8.64±0.22   | 7.71±0.23  | 6.65±0.71   | 8.57±1.55    | 6.84±0.92  | 13.97±0.53 | 35.45±0.79  |
| Asymmetric-20%| 59.36±0.36  | 57.97±0.24 | 59.76±0.53  | 56.11±0.60   | 58.58±0.51 | 58.89±0.40 | 65.26±0.76  |
| Asymmetric-30%| 51.06±0.44  | 49.86±0.54 | 49.53±0.79  | 47.12±0.73   | 49.04±0.91 | 51.46±0.15 | 56.97±1.22  |
| Asymmetric-40%| 42.49±0.23  | 41.51±0.67 | 40.62±0.79  | 38.98±0.54   | 39.72±0.76 | 41.96±0.92 | 47.62±0.79  |

Table 3: Average test accuracy (%) on CIFAR-100 over the last 10 epochs.

| Methods | best | last |
|---------|------|------|
| Standard CE | 82.68 | 81.10 |
| Decoupling | 79.22 | 78.24 |
| Co-teaching | 82.43 | 81.52 |
| Co-teaching+ | 50.66 | 48.52 |
| JoCoR | 82.82 | 81.06 |
| Co-learning | **82.95** | **82.18** |

Table 4: Classification accuracy (%) on Animal-10N dataset.

and last denotes the final test accuracy. Co-learning still outperforms other methods on the webly real-world noisy dataset, which demonstrates its robustness.

| Methods | best | last |
|---------|------|------|
| Standard CE | 84.50 | 83.86 |
| Decoupling | 85.53 | 85.28 |
| Co-teaching | 61.91 | 61.86 |
| Co-teaching+ | 81.61 | 81.24 |
| JoCoR | 77.94 | 77.86 |
| Co-learning | **87.57** | **86.56** |

Table 5: Classification accuracy (%) on Food-101N dataset.

4.3 Ablation study

To study the effect of our proposed components, the ablation study is conducted on CIFAR-10 with Symmetry-20% and Symmetry-50% noise, as shown in Figure 8.

Standard cross-entropy suffers from overfitting on noisy training data. Simply adding a self-supervision signal helps to a small extent but is not good enough. The main issue is the slow convergence of self-supervised learning, making the feature encoder $f$ quickly overfits label-dependent information but ignores feature-dependent information.

In order to slow down the inaccurate convergence of supervised learning, a data augmentation method, MixUp, is utilized to avoid early overfitting. As we can see, a balanced training that prompts both label-dependent information and feature-dependent information play crucial roles during the training procedure. Though Co-learning without structural similarity loss can obtain a good performance under the low noisy rate Symmetry-20%, it still overfits noisy training data under the high noisy rate Symmetry-50%. Co-learning utilizes the structural similarity between different pretext tasks to regularize supervised learning and keeps a robust noise-tolerant training procedure. These results show that self-supervised learning can help combat noisy labels, and a balanced training manner further regularizes the model from overfitting matters. Our proposed Co-learning is a more efficient paradigm to take advantage of feature-dependent information.
with both the intrinsic similarity and the structural similarity. Co-learning achieves robust training without trivial tricks, while other co-training-based methods need the small-loss trick and noisy rate estimation. Through extensive experiments on synthetic and real-world noisy datasets, we demonstrate that Co-learning outperforms other co-training-based methods and provides a new perspective to leverage self-supervised learning to assist robust deep learning. In the future, we will further explore the theoretical foundation and generalization analysis of Co-learning.

REFERENCES

[1] Eric Arazo, Diego Ortego, Paul Albert, Noel O’Connor, and Kevin Mcginnness. 2019. Unsupervised Label Noise Modeling and Loss Correction. In Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97), Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, 312–321. http://proceedings.mlr.press/v97/arazo19a.html

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. MixMatch: A Holistic Approach to Semi-Supervised Learning. In Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alch´e-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc., Vancouver. https://proceedings.neurips.cc/paper/2019/file/1ed1384b499a65f5b77e0e94bce27f7-Paper.pdf

[3] Valda Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101 – Mining Discriminative Components with Random Forests. In Computer Vision – ECCV 2014, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer International Publishing, Cham, 446–461.

[4] Mathilde Caron, Ishaan Misra, Julien Mairal, Praya Guoyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. Advances in Neural Information Processing Systems 33 (2020).

[5] Pengfei Chen, Ben Ben Liao, Gaoyong Chen, and Shengyu Zhang. 2019. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97), Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, 1062–1070. http://proceedings.mlr.press/v97/chen19g.html

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119), Hal Daumé III and Aarti Singh (Eds.). PMLR, Vienna, 1597–1607. http://proceedings.mlr.press/v119/chen20j.html

[7] Xinlei Chen and Kaiming He. 2021. Exploring Simple Siamese Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Virtual, 12125–12124. https://doi.org/10.1109/CVPR4.2021.01258

[8] Yifan Ding, Liqiang Wang, Deliang Fan, and Boqing Gong. 2018. A Semi-Supervised Two-Stage Approach to Learning from Noisy Labels. In 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, Lake Tahoe, 12125–12124. https://doi.org/10.1109/WACV.2018.8323089

[9] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. 2018. Unsupervised Representation Learning by Predicting Image Rotations. In International Conference on Learning Representations. ICLR, Vancouver, 1–16. https://openreview.net/forum?id=S1t4N20-

[10] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhouhan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. 2020. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., Virtual, 21217–21234. https://proceedings.neurips.cc/paper/2020/file/3fa460d0c4fcee70412b17b8192b255e-Paper.pdf

[11] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Xiao Mu, Wenhua Hu, Iqor Tsang, and Masashi Sugiyama. 2018. Robust training of deep neural networks with extremely noisy labels. In Advances in Neural Information Processing Systems, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc., Montreal. https://proceedings.neurips.cc/paper/2018/file/a19744e268754fb0148b6b17647355b7-Paper.pdf

[12] Juan-gan Han, Ping Luo, and Xiao-gang Wang. 2019. Deep Self-Learning From Noisy Labels. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, Seoul, 5138–5147.

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Virtual, 9729–9738.

[14] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using Trusted Data to Train Deep Networks on Labels Corrupted by Severe Noise.

Figure 8: Results of ablation study on CIFAR-10 with Symmetry-20% and Symmetry-50% noise.

4.4 Discussion

Though these co-training-based methods for combating noisy labels are based on the assumption that different views of data should provide extra valuable information, networks with the same architecture provide limited variety caused by random initialization. In general, Decoupling firstly leverages the co-training method with a cross-update technique based on disagreements from two different views. Inspired by the small-loss trick from MentorNet, Co-teaching cross-updates samples selected by the small-loss. Co-teaching combines disagreement strategy and the small-loss trick, which means add a disagreement selection step in Co-teaching. JoCoR simplifies the above ideas and jointly updates two networks still based on the small-loss trick and an extra Jensen-Shannon divergence between predictions from two peer networks.

Co-learning further predigests these co-training methods through a shared feature encoder with two exclusive heads, which provides different views of data. Besides, Co-learning explores a reasonable scheme that leverages unsupervised learning to assist robust deep learning.

5 CONCLUSION

In this paper, we propose Co-learning for combating noisy labels by utilizing both label-dependent information from supervised learning and feature-dependent information from self-supervised learning. Instead of training two classifiers, Co-learning trains two heads with a shared feature encoder and regularizes the model
