SELFIE: Refurbishing Unclean Samples for Robust Deep Learning

Hwanjun Song [1] **Minseok Kim** [1] **Jae-Gil Lee** [1]

Abstract

Owing to the extremely high expressive power of deep neural networks, their side effect is to totally memorize training data even when the labels are extremely noisy. To overcome overfitting on the noisy labels, we propose a novel robust training method called *SELFIE*. Our key idea is to selectively refurbish and exploit unclean samples that can be corrected with high precision, thereby gradually increasing the number of available training samples. Taking advantage of this design, *SELFIE* effectively prevents the risk of noise accumulation from the false correction and fully exploits the training data. To validate the superiority of *SELFIE*, we conducted extensive experimentation using four real-world or synthetic data sets. The result showed that *SELFIE* remarkably improved absolute test error compared with two state-of-the-art methods.

1. Introduction

As the size of available data sets increases rapidly, deep neural networks have achieved remarkable performance in numerous machine learning tasks, such as image classification (Krizhevsky et al., 2012) and object detection (Redmon et al., 2016). However, owing to the high capacity to fit any noisy labels, it is known that a small portion of mislabeled samples in training data can severely hurt the model performance. In particular, Zhang et al. (2017) have shown that a standard convolutional neural network can fit the entire training data with any ratio of noisy labels and eventually leads to poor generalization on the test data. Thus, the key issue is how to train the deep neural network robustly even when mislabeled samples exist within the training data.

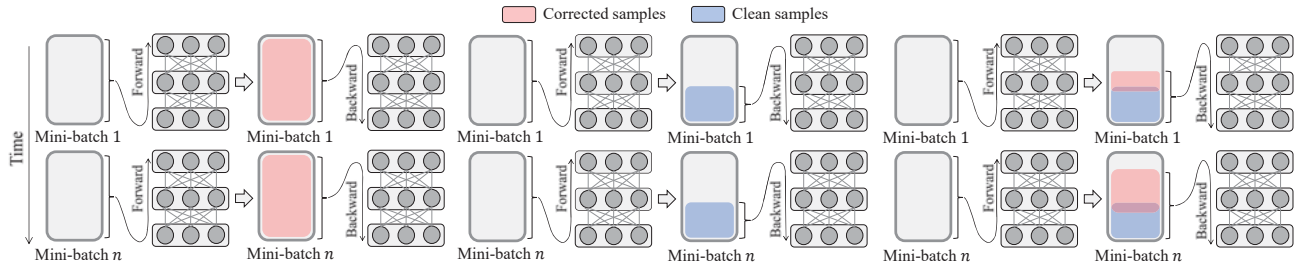
A typical method is using "loss correction" that corrects the loss of training samples based on the estimated noise transition matrix (Zhang et al., 2017; Goldberger & Ben-Reuven, 2017; Patrini et al., 2017; Chang et al., 2017). As shown in Figure 1(a), the forward or backward losses of *all* samples in each mini-batch are corrected and subsequently back-propagated to update the network. However, owing to the difficulty in estimating the noise transition matrix, it is inevitable that the network accumulates the error incurred by the *false* correction, especially when the number of classes or the number of mislabeled samples is large (Jiang et al., 2018; Han et al., 2018).

To be free of the *false* correction, many recent studies have adopted "sample selection" that filters out true-labeled samples from the training data (Kumar et al., 2010; Jiang et al., 2018; Han et al., 2018). They identified the clean samples out of the mini-batch based on their forward losses, and used them to update the network, as shown in Figure 1(b). In practice, Han et al. (2018) have shown that training on the clean samples yields a much better performance than correcting the entire sample on extremely noisy data. However, focusing on selected clean samples favors *easy* samples and thus ignores numerous useful *hard* samples, which make the network more accurate and robust (Shrivastava et al., 2016; Chang et al., 2017; Lin et al., 2018). Therefore, for a more robust training on noisy labels, we propose to *refurbish* unclean samples rather than just trash them in order to enable a *full* exploration of the training data.

In this paper, we propose a hybrid approach, called ***SELFIE*** *(SELectively reFurbIsh unclEan samples)*, that achieves the advantages of both "loss correction" and "sample selection". As stated above, loss correction allows for a full exploration of the training data by re-weighting all the losses; however, it suffers from the correction error. Conversely, sample selection effectively eliminates the noise accumulation by discarding all unclean samples but uses only the partial exploration of the training data. Thus, as illustrated in Figure 1(c), our key idea is to use *refurbishable* samples that can be corrected with a high precision, together with clean samples. Specifically, we *selectively* correct the losses of the training samples classified as refurbishable and combine them with the losses of clean samples to propagate backward. Because the precision of the correction highly depends on the network performance, the proportion of refurbishable samples increases gradually as the training step progresses, and eventually covers all samples in the training data. Overall,

[1]Graduate School of Knowledge Service Engineering, KAIST, Daejeon, Korea. Correspondence to: Jae-Gil Lee <jaegil@kaist.ac.kr>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.



(a) Loss Correction. (b) Sample Selection. (c) *SELFIE*.

Figure 1. Comparison of different robust training procedures: (a) shows the training procedures of *loss correction*, (b) shows the training procedures of *sample selection*, and (c) shows the training procedures of *SELFIE*.

Table 1. Comparison of state-of-the-art training methods with *SELFIE*: each method is grouped into three categories. In the first column, "Flexibility": need not be tied with a specific architecture; "No Pre-train": can be trained from scratch; "Heavy Noise": can combat the heavy noise; "Full Exploration": can use all samples in training data. (This table except the last "Hybrid" column is adapted from Han et al. (2018)'s work.)

Category	Loss Correction			Sample Selection			Hybrid
Method	*Bootstrap'15*	*F-correction'17*	*ActiveBias'17*	*Decouple'17*	*MentorNet'18*	*Coteaching'18*	***SELFIE***
Flexibility	×	○	○	○	×	○	○
No Pre-train	○	×	○	○	×	○	○
Heavy Noise	×	×	×	×	○	○	○
Full Exploration	○	○	○	×	×	×	○

SELFIE reduces the possibility of the false correction while exploiting the full training data.

We conducted extensive experiments to validate the superiority of *SELFIE*. DenseNet (Huang et al., 2017) and VGG-19 (Simonyan & Zisserman, 2014), which are two popular convolutional neural networks, were trained on not only simulated noisy CIFAR-10, CIFAR-100, and Tiny-ImageNet data sets, but also a real-world ANIMAL-10N data set. Compared with two state-of-the-art methods, *SELFIE* significantly (by up to $10.5pp$[1]) improves the robustness to the noisy data sets under any ratio of noisy labels.

2. Related Work

Recently, numerous studies have been performed to address the problem of learning from noisy labels. We categorize them into three groups along with *SELFIE*: loss correction, sample selection, and their hybrid. Table 1 systematically compares state-of-the-art methods for robust training.

Loss Correction: *Bootstrap* (Reed et al., 2015) trained the network using their own reconstruction-based objective to correct the usual prediction with the notion of perceptual consistency. *F-correction* (Patrini et al., 2017) pre-trained a normal network to estimate the probability of each class being corrupted into another, in order to re-weight the forward or backward losses resulting from noisy labels. *ActiveBias* (Chang et al., 2017) emphasized uncertain samples with high prediction variances; thus, the heuristically computed prediction variance was used to re-weight the backward losses of samples in the mini-batch. Ren et al. (2018) included small true-labeled validation data into the training data and re-weighted the backward losses of the mini-batch samples such that the updated gradient minimized the losses of those validation data. This family of methods operated well on moderately noisy data, but they failed to handle heavily noisy data owing to the inferiority of the accumulated error (Natarajan et al., 2013; Han et al., 2018).

Sample Selection: *Hard example mining* (Shrivastava et al., 2016) improved training convergence by removing easy samples, which tended to take a big portion of training data, to concentrate on learning hard samples. *Decouple* (Malach & Shalev-Shwartz, 2017) performs the decoupling of when to update from how to update. It maintained two networks and updated the networks using the samples with different label predictions. *MentorNet* (Jiang et al., 2018) pre-trained an additional teacher network to supervise the training of a student network. During training, the teacher network provides the student network with clean samples of which their labels are probably correct. *Coteaching* (Han et al., 2018) also uses two networks, but each network selects its small-loss samples as clean samples. Subsequently, each network feeds such clean samples to its peer network for further training. This family of methods achieved a much better performance on heavily noisy data by ignoring all unclean samples containing many mislabeled instances. However, at the same time, it is known that they also eliminate numerous useful samples for robust training (Shrivastava et al., 2016; Chang et al., 2017; Lin et al., 2018).

[1] A *pp* is the abbreviation of a percentage point.

To the best of our knowledge, *SELFIE* is the first method to satisfy the two conflicting factors: *heavy noise* and *full exploration*, by taking advantage of both categories. The concept of the refurbishable sample enables the noise accumulation to be minimized from mislabeled samples, as well as to exploit full exploration of training data. In Section 4.1, we have empirically verified that *SELFIE* outperforms *ActiveBias* and *Coteaching*, which are regarded as the state-of-the-arts of each category.

3. Robust Training via *SELFIE*

3.1. Overview

Let (x_i, y_i^*) be the pair of the sample x_i and its true label y_i^*, and $\mathcal{D} = \{(x_i, y_i^*)|1 \leq i \leq N\}$ be the training data set. However, sample labels are corrupted in many real-world classification tasks. We therefore assume that the mini-batch $\mathcal{M} = \{(x_i, \tilde{y}_i)|1 \leq i \leq b\}$ consists of the sample x_i with the label $\tilde{y}_i$ that may not be true, where $b \ll N$. Subsequently, in standard training, the parameter θ of the neural network is updated according to the descent direction of the expected loss on the mini-batch as in Eq. (1), where α and $\mathcal{L}$ are the given learning rate and loss function.

$$\theta_{t+1} = \theta_t - \alpha \nabla \Big(\frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \mathcal{L}(x, \tilde{y}; \theta_t) \Big) \tag{1}$$

In this study, we modify the update equation to render the network more robust on noisy labels. Let $\mathcal{C} \subseteq \mathcal{M}$ be the *clean* samples and $\mathcal{R} \subseteq \mathcal{M}$ be the *refurbishable* samples. We correct the backward loss of the refurbishable sample $x \in \mathcal{R}$ by replacing its corrupted label $\tilde{y}$ with the *refurbished* label y^{refurb}. Subsequently, as in Eq. (2), we back-propagate the losses for the refurbishable and clean samples to update the network. Here, it is *not* necessarily true that $\mathcal{R} \cap \mathcal{C} = \emptyset$. If a sample $x \in \mathcal{R} \cap \mathcal{C}$, being refurbishable precedes being clean because mislabeled instances could be included even in $\mathcal{C}$;[2] that is, the sample x needs to be refurbished.

$$\begin{aligned} \theta_{t+1} = \theta_t - \alpha \nabla \Big(\frac{1}{|\mathcal{R} \cup \mathcal{C}|} \big(\sum_{x \in \mathcal{R}} \mathcal{L}(x, y^{refurb}; \theta_t) \\ + \sum_{x \in \mathcal{C} \cap \mathcal{R}^{-1}} \mathcal{L}(x, \tilde{y}; \theta_t) \big) \Big) \end{aligned} \tag{2}$$

To update the network by Eq. (2), the key challenge is how to construct $\mathcal{R}$ as well as correct the loss of $x \in \mathcal{R}$, which will be discussed in the next section. On the contrary, there have been extensive studies on how to construct $\mathcal{C}$. Thus, for $\mathcal{C}$, we simply adopt the widely used *loss-based* separation method (Jiang et al., 2018; Han et al., 2018) that selects $(1 - \tau) \times 100\%$ of low-loss instances as clean samples, where τ is the noise rate. If τ is unknown, τ can be inferred using cross-validation (Liu & Tao, 2016; Li et al., 2017).

[2] The evidence that $\mathcal{C}$, in fact, has quite many mislabeled samples is presented in Section 4.1.4.

3.2. Main Concept: Selective Loss Correction

3.2.1. Criterion of Refurbishable

Interestingly, before the network fully fits the noisy labels, the label prediction of mislabeled samples either *(i)* changes inconsistently or *(ii)* corresponds to their *true* labels with high probability owing to the learner's perceptual consistency (Reed et al., 2015).

Hence, we aim to distinguish between the two cases to identify the samples that can be refurbished with high precision. Intuitively, the samples with consistent label predictions are regarded as refurbishable. The notion of being *refurbishable* is formalized as Definition 3.1 in which the predictive uncertainty uses the entropy to measure the consistency of label prediction.

Definition 3.1. A sample x is *refurbishable* if the predictive uncertainty in Eq. (3) $F(x; q) \leq \epsilon$ $(0 \leq \epsilon \leq 1)$.

$$F(x; q) = (1/\delta)\ \text{entropy}\big(P(y|x; q)\big) \quad \square \tag{3}$$

The definition of $F(x; q)$ is as follows. Let $\hat{y}_t = \Phi(x, \theta_t)$ be the predicted label of the sample x at time t and $H_x(q) = \{\hat{y}_{t_1}, \hat{y}_{t_2}, \ldots, \hat{y}_{t_q}\}$ be the label history of the sample x that stores the predicted labels of the previous q times, where Φ is a neural network. Next, $P(y|x; q)$ is formulated such that it provides the probability of the label $y \in \{1, 2, \ldots, k\}$ estimated as the label of the sample x based on $H_x(q)$ as in Eq. (4), where $[\cdot]$ is the Iverson bracket[3].

$$P(y|x; q) = \frac{\sum_{\hat{y} \in H_x(q)} [\hat{y} = y]}{|H_x(q)|} \tag{4}$$

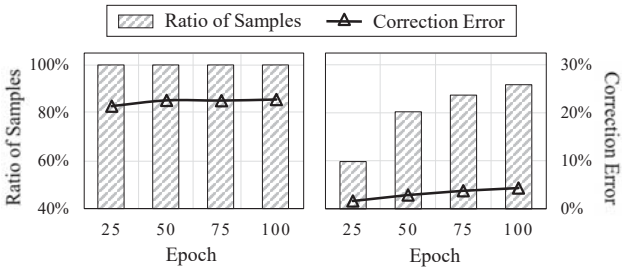
Then, to quantify uncertainty, we adopt the information entropy (Chandler, 1987) in Eq. (5). The predictive uncertainty $F(x; q)$ in Eq. (3) is now completed by Eq. (5).

$$\text{entropy}(P(y|x; q)) = -\sum_{j=1}^{k} P(j|x; q) \log P(j|x; q) \tag{5}$$

Because the uncertainty function $F(x; q)$ is bounded, we add the standardization term δ to re-scale the value to $[0, 1]$. For k classes, the minimum uncertainty is 0 when $P(m|x; q) = 1 \wedge \forall_{l \neq m} P(l|x; q) = 0$, and the maximum uncertainty is $-\log(1/k)$ when $\forall_j P(j|x; q) = 1/k$. Then, δ is defined as in Eq. (6).

$$\delta = -\sum_{i=1}^{k} (1/k) \log(1/k) = -\log(1/k) \tag{6}$$

[3] The Iverson bracket $[P]$ returns 1 if P is true; 0 otherwise.



(a) *Entire* Correction. (b) *Selective* Correction (Ours).

Figure 2. Analysis on *entire* and *selective* loss correction methods using DenseNet (L=25, k=12) on CIFAR-10 with symmetry noise 40%. The ratio of samples used for training (hatched bar) is plotted with the correction error (line).

3.2.2. Loss Correction

The loss of the refurbishable sample is corrected by replacing the corrupted label $\tilde{y}$ with the *refurbished* label y^{refurb} in Definition 3.2. Subsequently, it is combined with those of $(1-\tau)\times 100\%$ low-loss instances within the mini-batch, and back-propagated to update the network, as in Eq. (2). The leftover instances that might accumulate label noises are excluded from the update to pursue the robust learning.

Definition 3.2. A *refurbished* label y^{refurb} of the refurbishable sample x is the most frequently predicted label for previous q times, as in Eq. (7), where the sample x satisfies the condition $F(x;q)\leq\epsilon$.

$$y^{refurb} = \text{argmax}_{1\leq j\leq k} P(j|x;q) \quad \square \tag{7}$$

3.2.3. Quick Analysis

We peep at the experiment result to demonstrate the advantage of *selectively* correcting losses over *entirely* correcting losses. We trained DenseNet (L=25, k=12) using *SELFIE* on a noisy CIFAR-10 data set. For the entire correction method, we set ϵ to be 1 such that all samples were considered to be refurbishable regardless of their predictive uncertainty. For the selective correction method, we set ϵ as 0.05 such that only samples with low predictive uncertainty were considered to be refurbishable. As shown in Figure 2(a), the entire correction method uses all training samples during training, but it suffers from the high correction error on training samples of over 20%. That is, the network consistently accumulates the noise from mislabeled samples, thus causing poor generalization on test data. Conversely, as shown in Figure 2(b), the selective correction method reduces the noise by taking a part of training samples in the early stage of training (e.g., at the 25-th epoch), thereby achieving a significantly low correction error under 2%. Most importantly, even if only 60% of training samples are used for training initially, more training samples are added incrementally as the training epoch increases, while the low correction error is maintained at under 5%. Therefore, it is evident that the selective method guides the network to be trained more robustly on noisy data.

Algorithm 1 *SELFIE* Algorithm

INPUT: $\mathcal{D}$: data, $epochs$, γ: warm-up, τ: noise rate, ϵ: uncertainty threshold, q: history length, $restart$: # restarts
OUTPUT: θ_t: model parameter, $\mathbb{R}$: refurbished samples

1: $\mathbb{R}\leftarrow\emptyset$; /* $\mathbb{R}$ is entire refurbished samples in $\mathcal{D}$ */
2: **for** $r=0$ **to** $restart$ **do**
3: $t\leftarrow 1$;
4: $\theta_t\leftarrow$ Initialize the model parameter;
5: **for** $i=1$ **to** $epochs$ **do**
6: **for** $j=1$ **to** $|\mathcal{D}|/|\mathcal{M}|$ **do**
7: Draw a mini-batch $\mathcal{M}$ from $\mathcal{D}$;
8: **if** $i\leq\gamma$ **then** /* Warm-up periods */
9: /* Update by Eq. (1) */
10: $\theta_{t+1}=\theta_t-\alpha\nabla\big(\frac{1}{|\mathcal{M}|}\sum_{x\in\mathcal{M}}\mathcal{L}(x,\tilde{y};\theta_t)\big)$;
11: **else**
12: /* Clean sample selection */
13: $\mathcal{C}\leftarrow(1-\tau)\times 100\%$ of low-loss samples in $\mathcal{M}$;
14: /* Selective loss correction in Sec. 3.2 */
15: $\mathcal{R}\leftarrow\emptyset$; /* $\mathcal{R}$ is refurbished samples in $\mathcal{M}$ */;
16: **for each** $x\in\mathcal{M}$ **do**
17: **if** $F(x,q)\leq\epsilon$ **or** $x\in\mathbb{R}$ **do** /* By Eq. (3) */
18: $\mathcal{R}\leftarrow\mathcal{R}\cup(x,y^{refurb})$; /* Refurbish */
19: $\mathbb{R}\leftarrow\mathbb{R}\cup\mathcal{R}$; /* Aggregation */
20: /* Update by Eq. (2) */
21: $\theta_{t+1}=\theta_t-\alpha\nabla\Big(\frac{1}{|\mathcal{R}\cup\mathcal{C}|}\big(\sum_{x\in\mathcal{R}}\mathcal{L}(x,y^{refurb};\theta_t)+\sum_{x\in\mathcal{C}\cap\mathcal{R}^{-1}}\mathcal{L}(x,\tilde{y};\theta_t)\big)\Big)$;
22: $t\leftarrow t+1$;
23: **return** $\theta_t,\mathbb{R}$;

3.3. Algorithm Description

3.3.1. Main Algorithm: *SELFIE*

Algorithm 1 describes the overall procedure of *SELFIE*. First, during warm-up, by following the convention of the robust training, the network is trained on all training samples in the *default* manner, as shown in Eq. (1) (Lines 7–10). Even with the existence of noisy labels, deep networks learn clean and easy instances in the warm-up period without noise accumulation, which is known as *memorization* effect (Arpit et al., 2017; Jiang et al., 2018). Subsequently, after the warm-up period, $(1-\tau)\times 100\%$ of the low-loss samples are selected as clean samples $\mathcal{C}$ from the mini-batch $\mathcal{M}$ (Lines 11–13), and refurbishable samples $\mathcal{R}$ are identified and corrected by checking the condition for predictive uncertainty (Lines 14–18). Here, the refurbished samples $\mathcal{R}$ are aggregated for reuse (Line 19). After that, the network is updated based on the clean samples $\mathcal{C}$ along with the refurbished samples $\mathcal{R}$, as shown in Eq. (2) (Lines 20–21).

For more robust training, Algorithm 1 can be restarted multiple times with reusing the output $\mathbb{R}$ of the previous run (Line

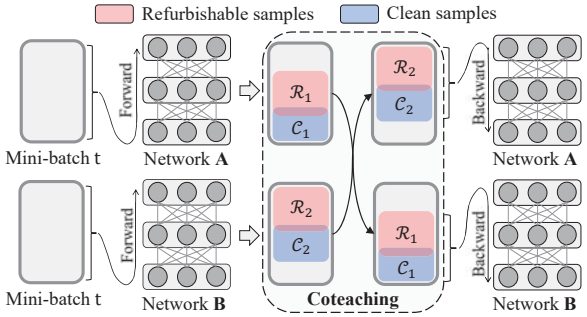


Figure 3. Training procedure of *CoSELFIE*.

1) as well as initializing the model parameter (Lines 2–4). This restart technique enables the network to be re-trained on *less-noisy* training data refurbished in the previous runs. In other words, a bunch of already refurbished samples are readily available from the very beginning of the current run. We demonstrate the effect of using the restart technique in Section 4.3.

3.3.2. Collaboration with *Coteaching*: *CoSELFIE*

An advantage of *SELFIE* is its *flexibility* with regard to collaboration with other orthogonal studies because it only needs a simple modification in the gradient descent step. Herein, for further improvement, we introduce *CoSELFIE* combined with *Coteaching* (Han et al., 2018), which is a state-of-the-art robust training algorithm. *CoSELFIE* maintains two networks simultaneously. In each mini-batch, each network identifies its own clean and refurbishable samples and feeds such samples to its peer network for further training, as demonstrated in Figure 3. A mini-batch t is provided to the network **A** and the network **B**; $\mathcal{R}_1$ and $\mathcal{C}_1$ are obtained from the network **A**, and $\mathcal{R}_2$ and $\mathcal{C}_2$ are obtained from the network **B**; for backpropagation, $\mathcal{R}_1$ and $\mathcal{C}_1$ are fed to the network **B**, and $\mathcal{R}_2$ and $\mathcal{C}_2$ are fed to the network **A**. It is known that *Coteaching* effectively removes the error incurred by the biased selection of training samples (Han et al., 2018). The advantage of *SELFIE* is boosted by *Coteaching*. We also demonstrate the improvement of *CoSELFIE* over *SELFIE* in Section 4.4.

4. Evaluation

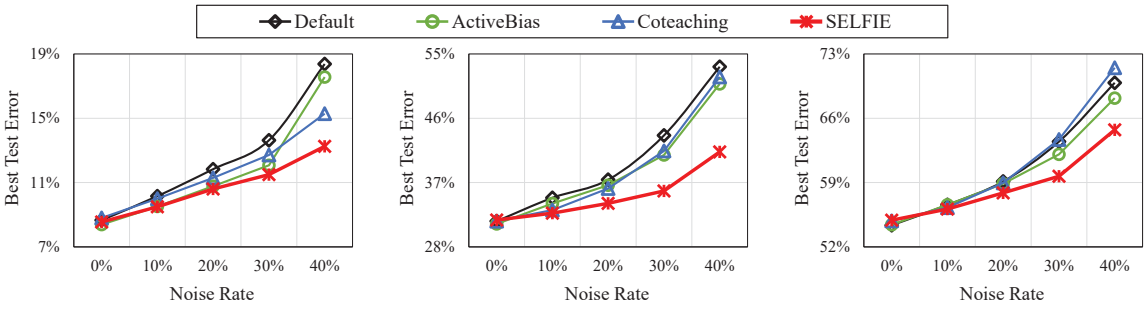
Data Sets: To validate the superiority of *SELFIE*, we performed an image classification task on *four* benchmark data sets: CIFAR-10 (10 classes)[4] and CIFAR-100 (100 classes)[4], classification of a subset of 80 million categorical images, with $50,000$ training and $10,000$ testing images; Tiny-ImageNet (200 classes)[5], classification of a subset of ImageNet (Krizhevsky et al., 2012), with $100,000$ training and $10,000$ testing images; ANIMAL-10N (10 classes)[6], our proprietary real-world noisy data set of human-labeled online images for 10 confusing animals, with $50,000$ training and $5,000$ testing images. Please note that, in ANIMAL-10N, noisy labels were injected *naturally* by human mistakes, where its noise rate was estimated at 8%. It has been released on our site[6], and its details can be found in Appendix B (supplementary material). We did not apply any data augmentation or pre-processing procedures.

Noise Injection: Except ANIMAL-10N, since all data sets are clean, we artificially corrupted these data sets using a typical method for the evaluation of noisy labels (Reed et al., 2015; Patrini et al., 2017; Han et al., 2018). As shown in Figure 7, for k classes, we applied the *noise transition matrix* $\mathbf{T}$: *(i)* ***pair noise***: $\exists_{j\neq i}\mathbf{T}_{ij} = \tau \wedge \forall_{k\neq i,k\neq j}\mathbf{T}_{ik} = 0$, and *(ii)* ***symmetry noise***: $\forall_{j\neq i}\mathbf{T}_{ij} = \frac{\tau}{k-1}$, where $\mathbf{T}_{ij}$ is the probability of the true label i being flipped to the corrupted label j and τ is the noise rate. It is known that pair noise is more realistic than symmetry noise because labelers may induce mistakes only within few classes (Han et al., 2018; Ren et al., 2018). To evaluate the robustness on varying noise rates from light noise to heavy noise, we tested five noise rates $\tau \in \{0.0, 0.1, 0.2, 0.3, 0.4\}$.

Network and Hyperparameters: For the classification task, we trained DenseNet (L=25, k=12) and VGG-19 with a momentum optimizer. Specifically, we used a momentum of 0.9, a batch size of 128, a dropout of 0.2 (Srivastava et al., 2014), and batch normalization (Ioffe & Szegedy, 2015). For the training schedule, following the experimental setup of Huang et al. (2017), we trained the network for 100 epochs and used an initial learning rate of 0.1, which was divided by 5 at 50% and 75% of the total number of epochs. Regarding the hyperparameters, we fixed $restart$ to 2 (i.e., restarted Algorithm 1 *twice* after the first run) and used the best uncertainty threshold $\epsilon = 0.05$ and history length $q = 15$, which were obtained from a grid $\epsilon = \{0.05, 0.10, 0.15, 0.20\}$ and $q = \{10, 15, 20\}$. (See Section 4.5 for details.) The warm-up threshold γ was set to 25 for the initial learning.

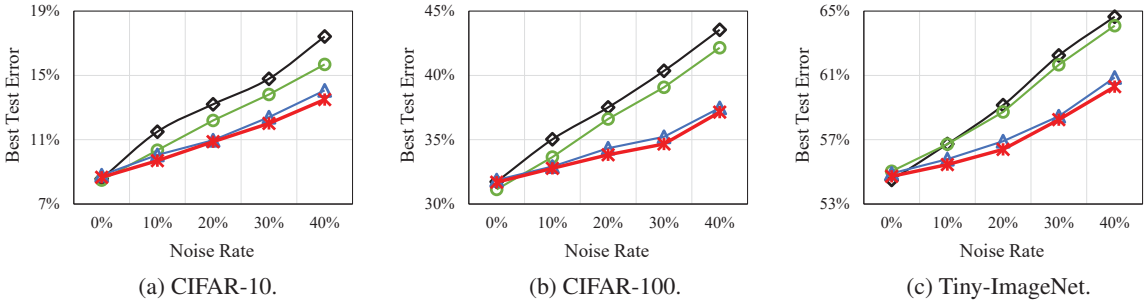
Algorithms: We compared *SELFIE* with a baseline algorithm (denoted by *Default*) and two state-of-the-art robust training algorithms. *Default* trains the network without any processing for the noisy labels. The others are the representatives of loss correction and sample selection strategies, respectively. *ActiveBias* (Chang et al., 2017) corrects the backward loss of training samples by prediction variance. *Coteaching* (Han et al., 2018) selects the clean samples by the loss-based separation and adopts the cotraining (Blum & Mitchell, 1998) mechanism. All the algorithms were implemented using TensorFlow $1.8.0$[7] and executed using a single NVIDIA Tesla V100 GPU. For reproducibility, we provide the source code at https://github.com/kaist-dmlab/SELFIE.

[4] https://www.cs.toronto.edu/~kriz/cifar.html
[5] https://www.kaggle.com/c/tiny-imagenet
[6] https://dm.kaist.ac.kr/datasets/animal-10n
[7] https://www.tensorflow.org/versions/r1.8



(a) CIFAR-10. (b) CIFAR-100. (c) Tiny-ImageNet.

Figure 4. The best test error of the four training methods using DenseNet on three data sets with varying **pair noise** rates.



(a) CIFAR-10. (b) CIFAR-100. (c) Tiny-ImageNet.

Figure 5. The best test error of the four training methods using DenseNet on three data sets with varying **symmetry noise** rates.

Table 2. The best test error (%) on **pair noise** 40% in Figure 4.

Method	CIFAR-10	CIFAR-100	Tiny-ImageNet
Default	18.4±0.35	53.2±0.46	69.9±0.28
ActiveBias	17.6±0.33	50.8±0.08	68.2±0.06
Coteaching	15.3±0.30	51.8±0.75	71.5±0.19
SELFIE	**13.2±0.06**	**41.3±0.15**	**64.7±0.08**

Table 3. The best test error (%) on **symmetry noise** 40% in Figure 5.

Method	CIFAR-10	CIFAR-100	Tiny-ImageNet
Default	17.4±0.25	43.6±0.21	64.7±0.21
ActiveBias	15.7±0.39	42.2±0.29	64.1±0.35
Coteaching	14.1±0.18	37.5±0.01	60.9±0.13
SELFIE	**13.5±0.04**	**37.1±0.02**	**60.3±0.08**

In support of reliable evaluation, we repeated every test *thrice* and reported the average and standard error of the best test errors. The test error at the end of the training is a common measure of robustness to noisy labels (Malach & Shalev-Shwartz, 2017; Jiang et al., 2018).

4.1. Performance Comparison

4.1.1. Result with Pair Noise

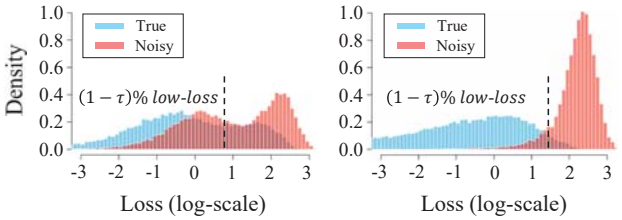
Figure 4 shows the test error of the four training methods using DenseNet (L=25, k=12) with varying pair noise rates. We present the results for VGG-19 in Appendix A (supplementary material) because of the lack of space, and the performance trends in the two architectures are similar with each other. Generally, at any noise rate, *SELFIE* achieved the lowest test error on all data sets. The difference in the test errors between *SELFIE* and other methods increased as the noise rate increased. In particular, at the heavy noise rate of 40%, *SELFIE* significantly reduced the *absolute* test error by 5.2*pp*–11.9*pp* compared with *Default*, 3.5*pp*–9.5*pp* compared with *ActiveBias*, and 2.1*pp*–10.5*pp* compared with *Coteaching*. The test errors of all methods at the pair noise rate of 40% are summarized in Table 2. Although the test errors of *ActiveBias* and *Coteaching* were lower than that of *Default*, they were not comparable to that of *SELFIE* except the light noise rates of 0%–20%. For a more robust training, this significant improvement in *SELFIE* proves that it is essential to *(i)* selectively correct the unclean samples and *(ii)* exploit the full exploration of the training data.

4.1.2. Result with Symmetry Noise

Figure 5 shows the test error of the four training methods using DenseNet (L=25, k=12) with varying symmetry noise rates. Similar to the pair noise, *SELFIE* generally outperformed other methods at any noise rate on all data sets. Quantitatively, at the heavy noise rate of 40%, *SELFIE* significantly reduced the *absolute* test error by 3.9*pp*–6.5*pp* compared with *Default*, 2.2*pp*–5.1*pp* compared with *ActiveBias*, and 0.4*pp*–0.6*pp* compared with *Coteaching*. The test errors of all methods at the symmetry noise rate of 40% are summarized in Table 3. Unlike the pair noise, *Coteaching* achieved a low test error comparable to *SELFIE*. In contrast, *ActiveBias* showed a slightly better performance than *Default*, but was significantly worse than *SELFIE* and *Coteaching*. Additionally, *Default* tended to show vulnerability even with the light noise rate of 10%.

Table 4. The best test errors (%) on ANIMAL-10N (8% noise).

Method	DenseNet (L=25, k=12)	VGG-19
Default	17.9±0.02	20.6±0.14
ActiveBias	17.6±0.17	19.5±0.26
Coteaching($\tau = 0.08$)	17.5±0.17	19.8±0.13
SELFIE($\tau = 0.08$)	**17.0±0.10**	**18.2±0.09**



(a) Pair Noise 40%. (b) Symmetry Noise 40%.

Figure 6. Histogram of the distributions of losses at the training accuracy of 50% on a noisy CIFAR-100 data set, where "True" and "Noisy" denote true-labeled samples and mislabeled samples.

4.1.3. Result with Realistic Noise

Table 4 summarizes the best test errors of the four training methods using the two architectures on ANIMAL-10N. In both architectures, *SELFIE* achieved the lowest test error. Specifically, *SELFIE* improved the *absolute* test error by up to 0.9*pp* using DenseNet (L=25, k=12) and 2.4*pp* using VGG-19. *SELFIE* maintained its dominance over other methods on *realistic* noise, though the performance gain was not that huge because of a light noise rate (i.e., 8%).

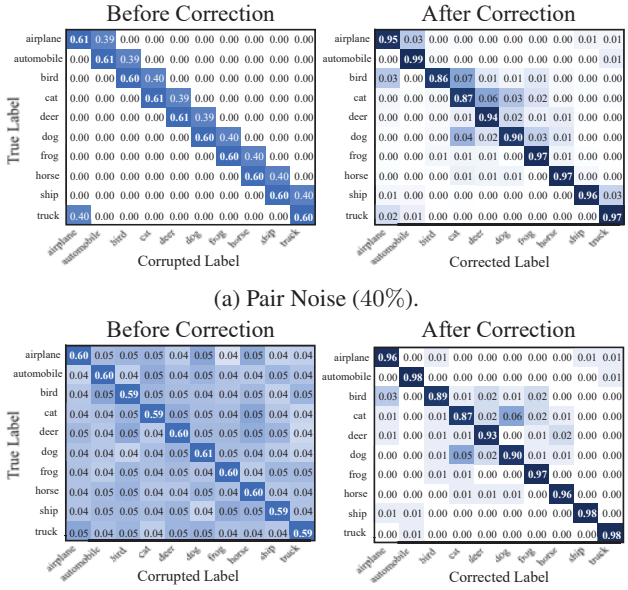
4.1.4. Anatomy of Loss-based Separation

An interesting observation is the considerable performance difference in *Coteaching* for pair and symmetry noises. The difference was found to be due to the loss-based separation proposed by *Coteaching*. As demonstrated in Figure 6(a), for the pair noise, the distribution of mislabeled samples was overlapped closely with that of true-labeled samples. That is, clean (i.e., $(1-\tau) \times 100\%$ low-loss) samples contained a significant number of mislabeled samples, thereby causing the network to accumulate the label noise. In contrast, for the symmetry noise in Figure 6(b), the two distributions were clearly separated. Most mislabeled samples exhibited a much higher loss than true-labeled samples.

SELFIE also adopts the loss-based separation to select clean samples, but achieved a remarkable performance for the pair noise, as shown in Figure 4, because even clean samples are regarded as refurbishable if their label does not conform to the most frequently predicted label as in Eq. (2). Consequently, the labels of mislabeled samples are refurbished with high precision, even when they are classified as clean.

4.2. Accuracy of Loss (Label) Correction

To verify the accuracy of the loss (label) correction, we show the confusion matrices before and after the correction in Figure 7. In the left column, the confusion matrices before correction correspond to the noise transition matrices. Many entries other than the diagonal entries have non-negligible probability because of pair or symmetry noises. In the right column, the confusion matrices after correction contain the refurbished labels determined by Definition 3.2. As opposed to the noise transition matrices, only few entries other than the diagonal entries have non-negligible probability. Although the noise rate was very high (40%), most of the diagonal entries had probability over 0.95, thus proving very high correction accuracy. Therefore, a large portion of noises was successfully cleared by *SELFIE*.



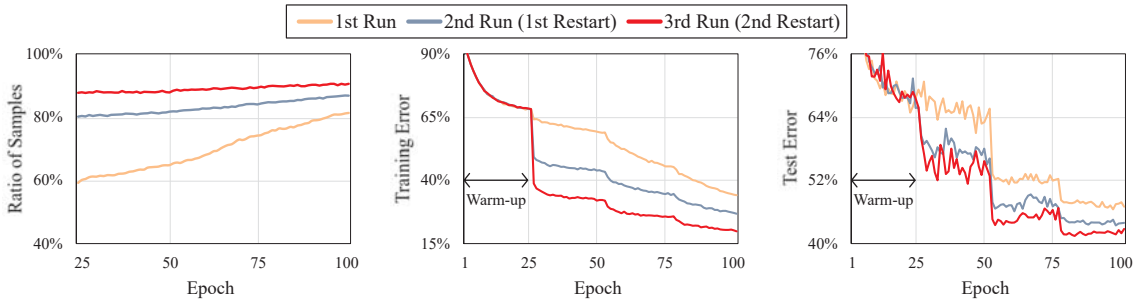
(a) Pair Noise (40%).

(b) Symmetry Noise (40%).

Figure 7. Confusion matrices on CIFAR-10 with (a) **pair noise** 40% and (b) **symmetry noise** 40%.

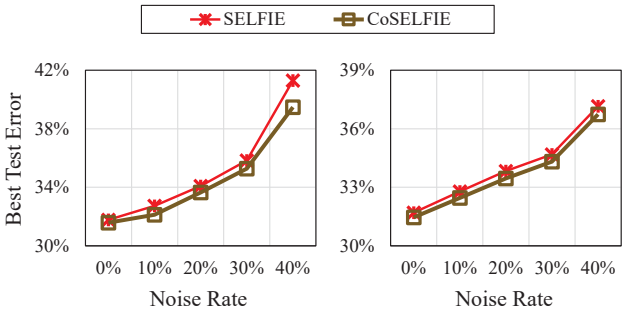
4.3. Performance Improvement by Restarts

Figure 8 shows the effect of the restart technique on CIFAR-100 with a pair noise of 40%. As shown in Figure 8(a), the number of samples available for training increased from 59.2% to 90.2% of the total training samples, as the number of runs increased. This effect encourages the network to be re-trained on a greater amount of training samples, which were refurbished in the previous runs. Therefore, the training and test errors were improved significantly in the next run, as shown in Figures 8(b) and 8(c). In detail, the training and test errors were 34.1% and 46.7% at the end of the first run, but were improved continuously through restart. They reached 26.8% and 43.4% at the end of the second run (i.e., first restart) and then 19.8% and 41.3% at the end of the third run (i.e., second restart). It is noteworthy that *SELFIE* achieved a significant reduction in *absolute* test error of 5.4*pp* using the restart technique. We expect that a larger number of restarts will further improve the performance of *SELFIE* at the expense of the training time.



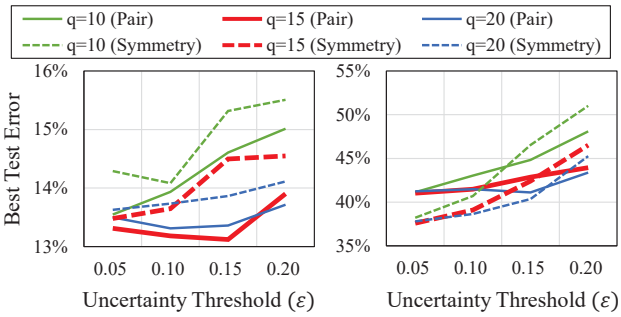
(a) Ratio of Samples. (b) Training Error. (c) Test Error.

Figure 8. Effect of restart on CIFAR-100 with **pair noise** 40%: (a) shows the ratio of samples used for training, (b) shows the reduction in training error, and (c) shows the reduction in test error.



(a) Pair Noise. (b) Symmetry Noise.

Figure 9. Performance improvement of *CoSELFIE* on CIFAR-100 with varying noise rates.



(a) CIFAR-10. (b) CIFAR-100.

Figure 10. Grid search on CIFAR-10 and CIFAR-100 with a noise rate of 40%.

4.4. Performance Improvement by Coteaching

Please recall that *CoSELFIE* (Section 3.3.2) is an extension of *SELFIE* based on *Coteaching* (Han et al., 2018). Figure 9 shows the test errors of *SELFIE* and *CoSELFIE* on CIFAR-100 with varying noise rates. Interestingly, by collaborating with *Coteaching*, the test error of *SELFIE* was further improved in both noise types. In particular, in the pair noise, the difference in the test errors between *SELFIE* and *CoSELFIE* tended to be larger as the noise rate increased. Quantitatively, compared with *SELFIE*, *CoSELFIE* reduced the *absolute* test error by $0.20pp$–$1.82pp$ in the pair noise and $0.24pp$–$0.41pp$ in the symmetry noise.

4.5. Hyperparameter Selection

SELFIE receives the two hyperparameters: the uncertainty threshold ϵ and the history length q. To decide the best hyperparameters, we trained DenseNet (L=25, k=12) on CIFAR-10 and CIFAR-100, each of which was corrupted by pair noise and symmetry noise at a rate of 40%. For the hyperparameter selection, the two hyperparameters were chosen in a grid $\epsilon \in \{0.05, 0.10, 0.15, 0.20\}$ and $q \in \{10, 15, 20\}$.

Figure 10 shows the test error of *SELFIE* obtained by the grid search on the two noisy data sets. Regarding the uncertainty threshold ϵ, lower test error was generally achieved with a smaller ϵ, because it induces that the more consistent samples from label predictions become the refurbishable samples. As for the history length q, the smallest q tended to yield the worst performance in the two data sets, regardless of the noise type. Although there is no clear winner between $q = 15$ and $q = 20$, the q value of 15 achieved the smallest test error when the ϵ value was the smallest at 0.05. Therefore, in all experiments, we set the uncertainty threshold ϵ to 0.05 and the history length q to 15.

5. Conclusion

In this paper, we proposed a novel method for robust training on noisy data, which we call *SELFIE*, that trains the network on precisely calibrated samples together with clean samples. Toward this goal, we introduced the concept of *selective loss correction* that identifies refurbishable samples and corrects their label with high precision. We conducted extensive experiments using two popular convolutional neural networks on four data sets with varying noise rates. Our experiment results showed that the robustness of a deep neural network on noisy data can be significantly improved by the selective loss correction on refurbishable samples. *SELFIE* guided the network to avoid noise accumulation from the false correction and allowed it to take advantage of the full exploration of training data. In addition, the results showed that the performance of *SELFIE* can be further improved by restarts and collaboration with other work. Overall, we believe that our work has greatly enhanced the robustness of deep learning on noisy data.

Acknowledgement

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No. 2017R1E1A1A01075927) and the MOLIT (The Ministry of Land, Infrastructure and Transport), Korea, under the national spatial information research program supervised by the KAIA (Korea Agency for Infrastructure Technology Advancement) (19NSIP-B081011-06).

References

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.

Blum, A. and Mitchell, T. Combining labeled and unlabeled data with co-training. In *COLT*, pp. 92–100. ACM, 1998.

Chandler, D. *Introduction to modern statistical mechanics*. Oxford University Press, 1987.

Chang, H.-S., Learned-Miller, E., and McCallum, A. Active Bias: Training more accurate neural networks by emphasizing high variance samples. In *NIPS*, pp. 1002–1012, 2017.

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

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. In *NIPS*, pp. 8536–8546, 2018.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *CVPR*, pp. 4700–4708. IEEE, 2017.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pp. 448–456, 2015.

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. 2309–2318, 2018.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *NIPS*, pp. 1097–1105, 2012.

Kumar, M. P., Packer, B., and Koller, D. Self-paced learning for latent variable models. In *NIPS*, pp. 1189–1197, 2010.

Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., and Li, L.-J. Learning from noisy labels with distillation. In *ICCV*, pp. 1910–1918, 2017.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Liu, T. and Tao, D. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):447–461, 2016.

Malach, E. and Shalev-Shwartz, S. Decoupling "when to update" from "how to update". In *NIPS*, pp. 960–970, 2017.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. Learning with noisy labels. In *NIPS*, pp. 1196–1204, 2013.

Patrini, G., Rozza, A., Menon, A. K., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pp. 2233–2241. IEEE, 2017.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. In *CVPR*, pp. 779–788. IEEE, 2016.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, 2015.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. In *ICML*, pp. 4334–4343, 2018.

Shrivastava, A., Gupta, A., and Girshick, R. Training region-based object detectors with online hard example mining. In *CVPR*, pp. 761–769. IEEE, 2016.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

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