Reproducibility Study of Self-Filtering : A Noise-Aware Sample Selection for Label Noise with Confidence Penalization

Kim Nayeon, Park Jeongwon, Choi bongjun
Department of Data Science
Seoul National University of Science and Technology
nayeon, bae301764, bongjun8@g.seoultech.ac.kr

Reproducibility Summary

Scope of Reproducibility

We examine the main claims of the original paper [1], which states that in an image classification task with noisy data. (i) SFT outperforms the almost state-of-the-art on CIFAR-10 with three noise types. (ii) The regularization term in the warming-up stage effectively separates clean and noisy samples. (iii) The regularization terms improve the performance of SFT. (iv) With a smaller T, SFT attains the best performance.

Additionally, we conduct experiments by modifying parts that are implemented differently from the paper. (i) Instead of considering only misclassified samples, the original implementation also takes into account the predicted probabilities within the memory bank. Therefore, we revise it to consider only misclassified samples as noise. (ii) We investigate whether the model can learn robustly even with changes in the data distribution.

Methodology

We utilized PyTorch to conduct additional experiments by reusing the authors' code along with additional code, and reimplemented scripts that are no longer supported. Further details are described in Section 3.3.

Results

Reproducing the SFT methodologies demonstrated high performance in various settings, establishing it as a robust method for learning with noisy labels. While the results generally showed superiority over other methodologies, not all experimental outcomes could be exactly replicated. Particularly, our implementation of the fluctuation methodology, based on the descriptions in the paper, showed lower performance than the original code provided by the authors. These findings suggest that while the method appears to be effective, discrepancies between the reproduced results and the authors' claims highlight the need for further experiments to validate its general applicability and robustness.

What was easy

The code in the original paper was made available, making it easy to use. The authors also clearly documented most of the hyperparameters that were used in the main experiments.

what was difficult

Different noise types required adjustments to the injection code, and the data loader's complexity posed comprehension difficulties. Discrepancies arose between reproduced results and those in the SFT paper's Table 1, complicating comparisons with other methodologies. Implementation differences further hindered result alignment, presenting a significant challenge in assessing SFT against other approaches.

33rd Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

1 Introduction

When deep learning models are trained on a label-noised dataset, the generalization performance would heavily degrade due to the presence of label noise.[2] Therefore, learning with noisy labels (LNL) poses great challenges for modern deep models.[3] Sample selection is an effective strategy to mitigate the effect of label noise in LNL. This strategy involves selecting and training on samples that have been correctly labeled (clean samples). Most recent LNL methods use sample selection based on the small loss criterion[4,5,6], leveraging the memorization effect[4]. However, the authors of the self-filtering paper point out issues with this approach. They highlight that when selecting samples showing small loss, noisy samples and boundary samples with high loss can be discarded together. Additionally, they note that the sample selection threshold, which needs to be specified, is a hyperparameter that is difficult to define beforehand.

Therefore, the authors propose a novel criterion called the Fluctuation criterion, which avoids selecting samples that transition from correct to misclassified (correct $\Rightarrow$ misclassified). They propose a memory bank module to store the prediction values for samples and a regularization term to avoid overconfidence. To demonstrate that their approach is effective and reliable, the authors performed several experiments. So we verified whether the authors' claims are indeed effective and reproducible.

2 Scope of reproducibility

The authors claimed to address the noisy label problem by using the fluctuation phenomenon and regularization term[1], and we conducted reproducibility experiments to verify this. The claims reproduced from the original paper through various experiments are as follows:

- Claim 1 : SFT outperforms the almost state-of-the-art on CIFAR-10 with three noise types. (Table 1)
- Claim 2 : The regularization term in the warm-up stage effectively separates clean and noisy samples. (Fig 7)
- Claim 3 : The regularization terms improve the performance of SFT. (Table 7, Fig 10)
- Claim 4 : With a smaller *T*, SFT attains the best performance. (Fig 9)

To verify whether the model can robustly learn despite changes in initial model parameters and data distribution, we conducted additional experiments by changing only the data seed while keeping the model seed constant. Upon reviewing the code provided in the paper, we found discrepancies between the provided code and the method described in the paper. This is explained in the following 2.

1. To verify whether the model can robustly learn despite changes in initial model parameters and data distribution, we conducted additional experiments by changing only the data seed while keeping the model seed constant.
2. The authors proposed fluctuation, where samples are considered noisy if they transition from correct predictions to incorrect predictions. However, upon examining the implementation in the open-sourced code, we found that selection is based on both the prediction probabilities stored in the memory bank and the presence of fluctuation. Therefore, we implemented the fluctuation criterion as described in the paper and conducted experiments.

Each experiment in Section 4 will support (at least) one of these claims.

3 Methodology

3.1 Model descriptions

Selecting with the Fluctuation Criterion The authors propose a new sample selection criterion, termed the fluctuation criterion, utilizing a dynamically updated memory bank, M, which stores the predictions from the last T epochs. The fluctuation criterion identifies a sample as fluctuated based on its classification consistency over time. Specifically, a fluctuation event is defined when a sample correctly classified in a previous epoch t_1 is misclassified in a subsequent epoch t_2, where $t_1 < t_2$ and both epochs belong to the set $\{t - T, \ldots, t\}$. The formula for this criterion is:

$$\beta = (\text{argmax}(p_{t1}) = y) \wedge (\text{argmax}(p_{t2}) = y), \tag{1}$$

Clean samples are those for which $\beta = 1$. These samples are identified and described as follows:

$$\tilde{D} = \{(x_i, y_i) \in D | \beta_i = 1\}_{i=1}^{n}. \tag{2}$$

These selected clean samples are subsequently utilized in further learning stages.

Warm-up This is a stage for learning the basic predictive capabilities. In this stage, we penalize the confidence of the network's output to prevent the model from learning too quickly. Under the pair-noise condition, if a sample is noisy, there is a possibility that class j (the second largest confidence of prediction) is the correct category. Therefore, regularization terms such as (3) and (4) are used. In this case, if $\alpha(P_j)$ is large, it can penalize the model when it shows overconfidence in class y. During the warm-up phase, the cross-entropy loss with the regularizer is used as the objective function.

$$R = -\alpha(p_j) \cdot \log p_j \tag{3}$$

$$\alpha(p_j) = \max(0, \Gamma - \frac{p_j}{p_y}) \tag{4}$$

$$\mathcal{L} = \mathbb{E}_{(x,y)\in\mathcal{D}} \left[\mathcal{L}_{CE}(f(x,\theta), y) + R(f(x,\theta), y)\right] \tag{5}$$

Main Learning By assigning weights to the misclassified classes in the loss function, the model can avoid overconfidence in correct predictions. This regularization term functions similarly to label smoothing by utilizing the model's predicted values. However, unlike label smoothing, which requires a predefined epsilon (7), the proposed regularization term is adaptively computed (6). Thus, it has the advantage of not requiring separate hyperparameter tuning. Through this regularization term, the model can be penalized for overconfidence, adaptively minimizing the expected loss for each class. Let $\tilde{D}$ denote the selected samples; the loss function for the main learning stage can be formulated as in (8).

$$\mathcal{L}_{CR} = -\frac{1}{K} \sum_{k\in[K]} \alpha(p_k) \cdot \log p_k \tag{6}$$

$$\mathcal{L}_{LS} = -\log p_y - \sum_{k\in[K]} \epsilon \log p_k \tag{7}$$

$$\mathcal{L} = \mathbb{E}_{(x,y)\in\tilde{D}} \left[\mathcal{L}_{CE}(f(x,\theta), y) + \lambda\mathcal{L}_{CR}(f(x,\theta), y)\right] \tag{8}$$

3.2 Datasets

Original paper experiments utilized CIFAR-10[7], CIFAR-100[7], and Clothing-1M[8] datasets. For simplification of the experiment, our study focused on CIFAR-10, a popular dataset with 60,000 32x32 color images across 10 classes, split into 50,000 training and 10,000 test images with equal samples per class.

3.3 Noisy Label Setting

To simulate the actual noise condition in real-world, we refer to [9] manually construct three noise types: symmetric, pair and instance-dependent label noise. Symmetric noise refers to noise added to data where incorrect labels are assigned with equal probability to each class. Pair noise refers to noise in data where incorrect labels are systematically assigned to a specific incorrect class, typically a neighboring or similar class. Instance noise refers to noise added to data at the individual instance level, where certain data points have incorrect labels or corrupted feature values, without a uniform pattern across the dataset.

3.4 Experimental setup and code

Following the settings of the original paper, we used the ResNet-18 model[10], with a batch size of 32 and momentum of 0.9, employing the Stochastic Gradient Descent (SGD) optimizer with a weight decay of 5e-4. The number of epochs was set to 75, with 10 epochs in the warming-up stage. The initial learning rate was 0.02, adjusted using the MultiStepLR scheduler to decrease by a factor of 0.1 at the 60th epoch. Unless specified otherwise for comparison experiments, the baseline was set with *a memory bank size T=3* and *confidence threshold* Γ*=0.2*.

We used the original code provided by the authors (https://github.com/1998v7/Self-Filtering), with modifications made for reproducibility experiments available at the following URL: https://github.com/BBongjun/Self-Filtering. The performance of the experiments was evaluated based on accuracy, with the F1 score of sample selection also considered. We conducted repeated experiments, recording the mean accuracy and standard deviation.

3.5 Computational requirements

The experiments were conducted on a system with an NVIDIA GeForce RTX 4090 GPU, AMD Ryzen Thread ripper PRO 5955WX 16-Cores CPU, and 125GB of RAM. Using these resources, running 75 epochs on the CIFAR-10 dataset took approximately 1 to 1.5 hours.

4 Results

4.1 Results reproducing original paper

4.1.1 Test accuracy on CIFAR-10 (Table 1. Result)

	SFT		Co-teaching		JoCor	
Condition	**Reported**	**Ours**	**Reported**	**Ours**	**Reported**	**Ours**
Symm. 20%	92.57 ± 0.32	92.59 ± 0.17	87.16 ± 0.11	82.73 ± 0.15	88.69 ± 0.19	85.11 ± 0.15
Symm. 40%	89.54 ± 0.27	90.34 ± 0.10	83.59 ± 0.28	78.35 ± 0.31	85.44 ± 0.29	80.40 ± 0.55
Pair 20%	91.53 ± 0.26	93.43 ± 1.59	86.91 ± 0.37	82.86 ± 0.17	87.75 ± 0.46	83.08 ± 0.47
Pair 40%	89.93 ± 0.47	89.66 ± 0.06	82.77 ± 0.57	76.30 ± 0.66	83.91 ± 1.49	62.94 ± 0.85
Inst. 20%	91.41 ± 0.32	91.97 ± 0.41	86.54 ± 0.11	82.65 ± 0.09	87.31 ± 0.27	83.19 ± 0.18
Inst. 40%	89.97 ± 0.49	89.08 ± 0.44	80.98 ± 0.39	74.99 ± 0.36	82.49 ± 0.57	68.67 ± 0.58

Table 1: Comparison of different methods under varying noise conditions with reported and our results.

The authors identified issues with the small loss criterion-based sample selection methods, such as Co-teaching[4] and JoCoR[6], and these were set as comparative baselines. In the original paper's code, the noisy labels were set to change continuously. To ensure consistency across experiments, we modified the code to use the same data for each experiment. The comparative baselines were tested using the hyperparameters specified in their respective original papers. The results of SFT were similar to those reported in the paper. However, the reproduced performance of Co-teaching and JoCoR was lower than the performance reported in their respective papers. Upon reviewing the original papers, we found that the performance we reproduced matched the results in those papers. This discrepancy is likely due to differences in hyperparameter settings used by the authors when experimenting with Co-teaching and JoCoR.

4.1.2 Effect of regularization term in Warm-up stage (Fig.7 Result)

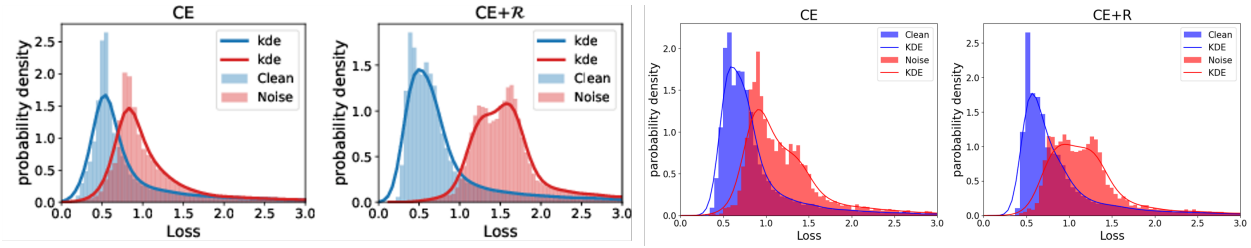

Figure 1: The loss distribution of noisy CIFAR-10 with Pair 40% after warming up [Reported(Left), Ours(Right)]

Under the pair noise condition, the authors claimed that the regularization term used during the warm-up phase helps prevent overconfidence of noise labels, effectively separating clean samples from noisy ones. Our experimental results did not show an obvious disparity between clean and noisy samples. However, we observed a slight effect of the regularization term in reducing overconfidence in noisy labels, as evidenced by a decrease in kurtosis.(Figure. 1)

4.1.3 Ablation study of each regularization term (Table 7, Fig. 10 Result)

	Pair. 40% Acc		Pair. 40% F-score		Inst. 40% Acc		Inst. 40% F-score	
Condition	**Reported**	**Ours**	**Reported**	**Ours**	**Reported**	**Ours**	**Reported**	**Ours**
SFT	89.74	88.53	0.963	0.956	90.06	89.22	0.969	0.946
w/o R	88.79	89.63	0.952	0.956	88.83	89.11	0.961	0.946
w/o L_{cr}	87.14	89.60	0.944	0.955	87.36	88.96	0.948	0.943
w/o R & L_{cr}	86.05	87.87	0.940	0.942	86.31	88.82	0.946	0.944

Table 2: Ablation study of each components. The results of test accuracy and F-score on variant noise labels with reported and our results.

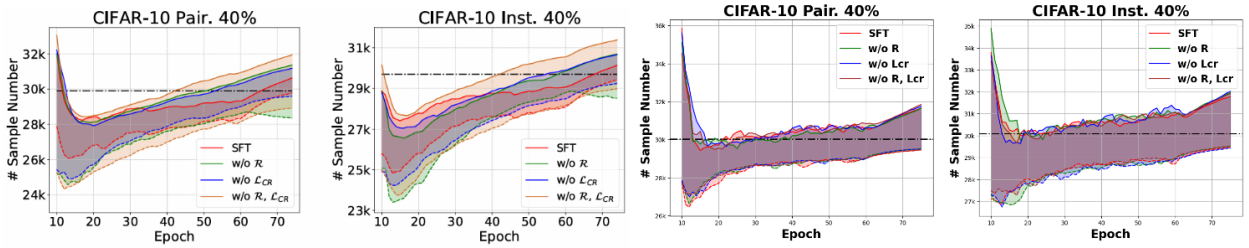

Figure 2: Ablation study of selected samples number. The solid lines and dashed lines de note the number of selected set $\tilde{D}$ and the clean labels in $\tilde{D}$, respectively. The horizontal dashed line denotes the actual clean samples number.[Reported(Left), Ours(Right)]

To verify the effect of the regularizer, we reproduced performance comparisons with and without the regularizer. According to the paper, SFT with the regularizer should show significantly higher performance. However, we did not observe a clear performance difference. Under the pair noise condition at 40%, there was a 1-2% performance difference compared to the results without SFT and the regularizer, but under the instance noise condition at 40%, there was almost no difference. Consequently, the results in Fig 10 were also not reproducible. Therefore, it is difficult to conclude that the regularizer has a significant effect.

4.1.4 Study of Memory bank Size (Fig. 9)

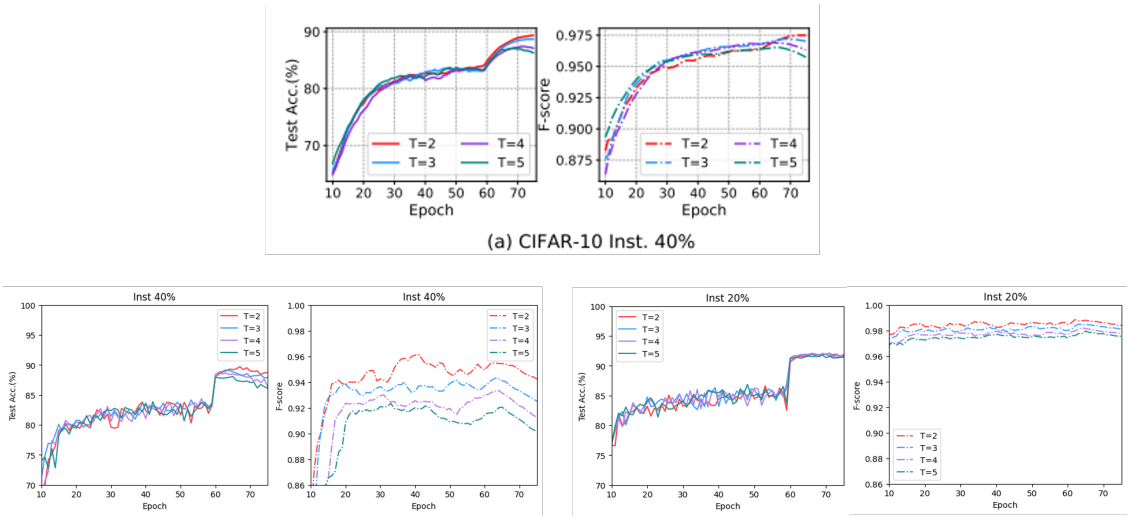

Figure 3: The hyper-parameter selection of memory bank size T. Test accuracy (%) and F-score are reported. [Reported (Top), Ours (Bottom) for instance 40% and 20%]

The authors mentioned that smaller memory bank sizes T yielded better performance, leading them to adopt T=3 for their experiments. To verify this, we varied the bank size and conducted experiments, including an instance 20% noise setting for comparison. Our results confirmed that smaller T values generally produced better performance, especially in 40% noise environments. However, while our F1 score trends were consistent with the original paper, our experiments showed a decreasing trend in F1 scores over epochs, contrary to the original paper's gradual increase.

4.2 Results beyond original paper

4.2.1 Robustness in various data distribution

To explore how performance adapts under different data distributions, we modified the random seed during data noise injection. Reported results correspond to those from Table 1, where experiments were repeated with consistent data and model seeds. In our approach, we varied the data seed while maintaining a fixed model seed to assess performance under different data distributions. Although performance levels fluctuated depending on the method and ratio of noise injection, the proposed approach maintained comparable performance. These additional experiments highlight the model's resilience to variations in data distribution, suggesting the robustness of their methodology.

Method	Instance		Symmetric		Pair	
	20%	40%	20%	40%	20%	40%
Reported	91.41 ± 0.32	89.97 ± 0.49	92.57 ± 0.32	89.54 ± 0.27	91.53 ± 0.26	89.93 ± 0.47
Ours	92.38 ± 0.16	89.99 ± 0.29	90.50 ± 0.27	92.11 ± 0.89	92.48 ± 0.13	89.67 ± 1.3

Table 3: Experimental Results under varying noise conditions with reported and our results.

4.2.2 Study about our fluctuation implementation based on paper

Method	Symmetric		Pair		Instance	
	20%	40%	20%	40%	20%	40%
Reported	91.41 ± 0.32	89.97 ± 0.49	92.57 ± 0.32	89.54 ± 0.27	91.53 ± 0.26	89.93 ± 0.47
Ours	90.88 ± 0.22	87.18 ± 0.15	92.55 ± 0.30	89.86 ± 0.31	91.57 ± 0.20	87.80 ± 0.42

Table 4: Ablation study about fluctuation implementation. The results of test accuracy and F-score on variant noise labels with reported and our results.

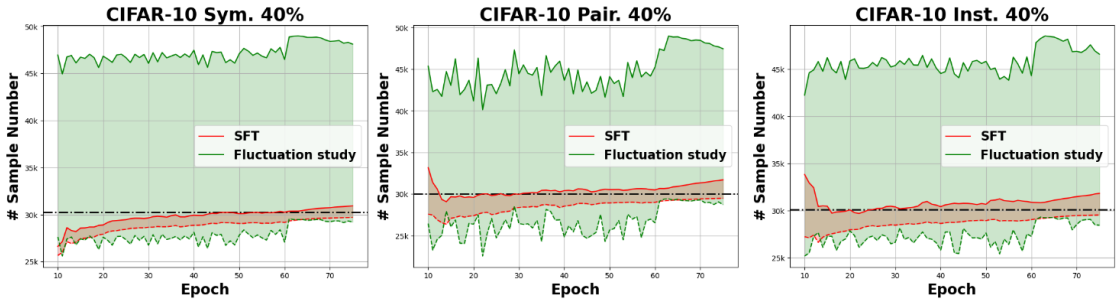

Figure 4: Ablation study of selected samples number. The solid lines and dashed lines de note the number of selected set $\tilde{D}$ and the clean labels in $\tilde{D}$, respectively. The horizontal dashed line denotes the actual clean samples number.

Upon implementing the fluctuation-based sample selection method described in the paper, we observed similar performance to SFT (Table 4) only under the Pair setting. However, under the Symmetric and Instance settings, the performance degraded. Upon examining the comparison graphs (Figure. 4) with SFT, it appeared that more noisy samples were selected, leading to learning from noisy labels, which seemed to be the cause of performance degradation. Hence, we suggest additional clarification regarding the fluctuation method implemented in the publicly available code.

5 Discussion

When reproducing the SFT methodologies using the provided code, it showed high performance in various environments and found to be one of the excellent noise label learning methods. However, when reproducing various experiments under the same conditions, the overall performance was reproduced, we could not achieve identical results in some experiments. In particular, when implementing fluctuation based on the paper's methodology, the performance was lower compared to the performance based on the code provided by the authors. Therefore, we thought that additional explanations from the authors are necessary to address this discrepancy.

5.1 What was easy and what was difficult

The availability of the publicly accessible code was advantageous, and we express our gratitude to the authors for making it available. However, it was somewhat difficult to fix the parts not implemented as described in the paper. Each comparison methodology used different types of noise, requiring us to adjust the noise injection code. Additionally, the complexity of the data loader's code structure made comprehension challenging. Discrepancies emerged between our reproduction results and those documented in Table 1 of the SFT paper, and the provided code's implementation diverged from the paper's assertions. Consequently, determining the appropriate experimental results for comparing SFT with other methodologies posed a significant challenge.

References

[1] Q. Wei, et al., "Self-filtering: A noise-aware sample selection for label noise with confidence penalization," in European Conference on Computer Vision, Cham, Switzerland: Springer Nature Switzerland, 2022.

[2] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, et al., "A closer look at memorization in deep networks," in Proceedings of the 34th International Conference on Machine Learning, 2017, pp. 233–242.

[3] H. Sun, C. Guo, Q. Wei, Z. Han, and Y. Yin, "Learning to rectify for robust learning with noisy labels," Pattern Recognition, vol. 124, p. 108467, 2022.

[4] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Coteaching: Robust training of deep neural networks with extremely noisy labels," in Advances in Neural Information Processing Systems, 2018.

[5] L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in International Conference on Machine Learning, 2018.

[6] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," in IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[7] A. Krizhevsky, G. Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[8] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[9] Y. Bai and T. Liu, "Me-momentum: Extracting hard confident examples from noisily labeled data," in IEEE International Conference on Computer Vision, 2021.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in IEEE Conference on Computer Vision and Pattern Recognition, 2016.