NFTT: Two-Stage Approach for Image Classification with Noisy Labels

Shuo He [hes11]
Department of Computing and Software
McMaster University
Hamilton, Canada
hes11@mcmaster.ca

Ziqing Xu [xuz69]
Department of Computing and Software
McMaster University
Hamilton, Canada
xuz69@mcmaster.ca

Abstract

Recent studies have shown that deep neural networks perform exceptionally well in a variety of applications, including image classification; nevertheless, this success depends on a large amount of correctly labeled training data. [1] Obtaining well-labeled training data by annotating labels manually is impractical as it is expensive and time-consuming. On the other hand, obtaining training data with noisy labels from online search engines and crowdsourcing platforms is relatively easy, fast, and inexpensive. The problem with deep neural networks is that they are prone to overfitting noisy, which degenerates performance. Therefore, in this project, we propose a two-stage approach, NFTT, to robustly train deep neural networks using such noisy labels and conduct an experiment on two noisy datasets to validate whether our proposed approach can outperform some existing methods.

Keywords: Image classification, Noisy labels

I. INTRODUCTION

An instance is regarded as noisy data if its label differs from the ground-truth label, which will inevitably deteriorate the robustness of the trained model, particularly for deep neural networks. [2] There are some existing methods for combating the noisy labels, such as sample selection methods (i.e., MentorNet, Decoupling, Co-Teaching, etc.), label correction, semi-supervised Learning, regularization, and noise-tolerant loss function.

Intuitively, better performance can be obtained as the training data becomes less noisy. [2] In this project, we propose a two-stage sample selection method, which is like Co-teaching. In other words, our proposed method could be considered an upgraded version of Co-teaching. Specifically, we first pre-train a deep neural network as a noise filter to remove samples that are more likely to be noisy samples and then use the filtered data points as input to train three deep neural networks simultaneously, namely, Tri-Teaching. The details of our proposed method, **NFTT** (**N**oise **F**ilter + **T**ri-**T**eaching), will be discussed in section 3.

Memorization effect and Co-Teaching are the two key components that motivate our proposed method. The experimental results (details in section IV) on real-world noisy

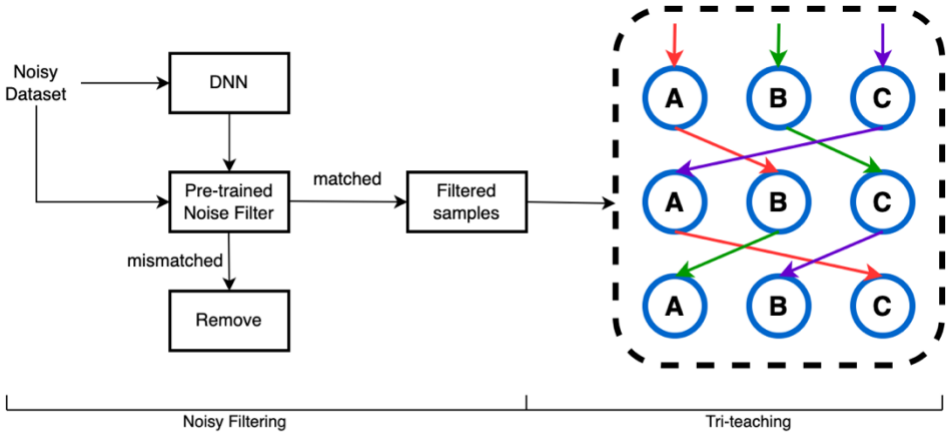

Figure 2. Overall pipeline of the proposed method, NFTT

datasets ANIMAL-10N and the three noise levels of CIFAR-10 demonstrate that NFTT outperforms Co-teaching in most tasks. The main contribution is that the proposed method performs well in the case of extremely high noise rate.

Our Pytorch implementation of NFTT is available at https://github.com/xuziqing98/CAS771_NFTT.

II. Related Work

A. *Memorization Effect*

Recent studies on the *memorization effect* have shown that deep neural networks would initially learn training data with clean labels in the early stage of training before they start memorizing noisy labels and eventually overfitting. [3] Therefore, the *memorization effect* suggests that the early learning phase is a promising time to detect training data that are more likely to be correctly labeled, which motivates the first stage (details of the noise filter will be discussed in Section III) of our proposed method.

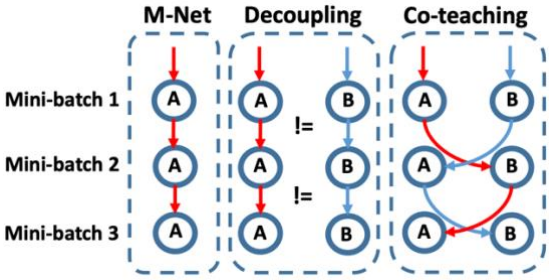

Figure 1. Error flow of MentorNet, Decoupling, and Co-teaching

B. *Co-teaching* [2]

Co-teaching is a sample selection method proposed by Bo et al. for handling the noisy labels. The key idea of Co-teaching is to maintain two networks simultaneously, each of which will back-propagate small loss samples selected by its peer network (these samples are considered to be potentially correctly labeled data) and update its parameters in each mini-batch during training.

The robustness of Co-teaching depends on the framework of the model, which can be explained by comparing it with two other sample selection methods, MentorNet [4] and Decoupling [5]. Errors from each network will be accumulated during the iteration of each mini-batch in MentorNet and Decoupling, as shown in Figure 1. However, in co-teaching, errors from one network will be transferred to its peer networks, each of which can attenuate different types of noisy labels due to its different learning capabilities. [2]

The idea of Tri-teaching stems from the fact that diverse learning capacities allow each network to filter distinct sorts of noisy labels. The details of Tri-teaching will be discussed in Section III.

III. Methodology

In this section, we describe our two-stage sample selection method for learning noisy labels. Figure 2 shows the pipeline of our proposed method, consisting of two stages, noise filtering,

and Tri-teaching. The Noise Filter is utilized to obtain the possibly clean data from the original noisy dataset that will be fed into the three networks for the training in the Tri-teaching stage. Therefore, the proposed algorithm is named NFTT, **N**oise **F**ilter + **T**ri-**T**eaching, (Algorithm 1). We will introduce each component of the proposed method in the following sections.

A. Stage 1: Noise Filter

Intuitively, as the training data becomes less noisy, higher performance may be gained. [2] Hence, the first stage aims to retrieve those data points that tend to be correctly labeled from the original noisy dataset. The selected data with possible clean labels will be provided as input for the training of Tri-teaching.

As lines 2 to 8 in Algorithm 1, we pre-train a deep neural network, named Noise Filter, using the noisy training dataset. By exploiting the *memorization effect*, the noise filter will first learn clean data in the early learning phase so that it is used to mine the data points with possibly clean labels for the second phase.

Specifically, as lines 9 to 13 in Algorithm 1, we use a pre-trained Noise Filter to remove data points that are potentially incorrectly labeled, for which the predictions of the Noise Filter are different from the original labels. The remaining dataset deemed less noisy will be used in the next stage to train three networks simultaneously.

B. Stage 2: Tri-teaching

Instead of training two networks, as in Co-teaching [2], we propose to train three deep neural networks simultaneously. Each network will choose its small loss samples as helpful information to instruct one of its peer networks for updating parameters in each mini-batch, as shown in Figure 2.

Specifically, we will maintain three networks a (with parameters w_a), b (with parameters w_b), and c (with parameters w_c). In each mini-batch data $\bar{D}$, each network will first select small loss samples of $R(T_t)$ percentage from $\bar{D}$ to remember, as lines 18 to 20 in Algorithm 1. Each network will then back-propagate the selected instances by one of its peer networks (switching from mini-batches to mini-batches) and update its parameters, as lines 21 to 29 in Algorithm 1. Namely, when the mini-batch index is even, network a (resp. b and c) will impart useful small loss instances to network b (resp. c and a), and when the batch index is odd, network a (corresponding to b and c) will impart those to the network c (resp. a and b). The Tri-teaching part of Figure 2 clearly illustrated this procedure.

We will update the remember rate $R(T_t)$ after every training epoch, the details of $R(T_t)$ will be explained in Section IV.

Intuitively, different classifiers are expected to have different learning abilities to filter out different types of noisy labels, which motivates us to exchange small loss samples. [2] In Tri-teaching, we would like to see if the performance can be improved by adding another classifier.

Algorithm 1 NFTT Algorithm

1: **Input** w_f, w_a, w_b, and w_c, learning rate η, fixed τ, exponent T_k, epoch $T_{f_{max}}$ and $T_{t_{max}}$, iteration $N_{f_{max}}$ and $N_{t_{max}}$
2: **for** T_f = 1, 2, ... , $T_{f_{max}}$ **do**
3: **Shuffle** training set D;
4: **for** N_f = 1, 2, ... , $N_{f_{max}}$ **do**
5: **Fetch** mini-batch $\bar{D}$ from D;
6: **Update** $w_f = w_f - \eta \nabla \mathcal{L}\ (f, \bar{D})$
7: **end for**
8: **end for**
9: **for** i = 1, 2, ... , $|D|$ **do**
10: **if** $f(D_i) \neq$ label of D_i **then**
11: **Update** $D = D - D_i$
12: **end if**
13: **end for**
14: **for** T_t = 1, 2, ... , $T_{t_{max}}$ **do**
15: **Shuffle** filtered training set D;
16: **for** N_t = 1, 2, ... , $N_{t_{max}}$ **do**
17: **Fetch** mini-batch $\bar{D}$ from D;
18: **Obtain** $\bar{D}_a = \arg\min_{D':|D'|\geq R(T_t)|\bar{D}|} \mathcal{L}(a, D')$
19: **Obtain** $\bar{D}_b = \arg\min_{D':|D'|\geq R(T_t)|\bar{D}|} \mathcal{L}(b, D')$
20: **Obtain** $\bar{D}_c = \arg\min_{D':|D'|\geq R(T_t)|\bar{D}|} \mathcal{L}(c, D')$
21: **if** N_t % 2 == 0 **then**
22: **Update** $w_a = w_a - \eta \nabla \mathcal{L}\ (a, \bar{D}_c)$
23: **Update** $w_b = w_b - \eta \nabla \mathcal{L}\ (b, \bar{D}_a)$
24: **Update** $w_c = w_c - \eta \nabla \mathcal{L}\ (c, \bar{D}_b)$
25: **else**
26: **Update** $w_a = w_a - \eta \nabla \mathcal{L}\ (a, \bar{D}_b)$
27: **Update** $w_b = w_b - \eta \nabla \mathcal{L}\ (b, \bar{D}_c)$
28: **Update** $w_c = w_c - \eta \nabla \mathcal{L}\ (c, \bar{D}_a)$
29: **end if**
30: **end for**
31: **Update** $R(T_t) = 1 - \min\{\frac{T_t}{T_k}\tau, \tau\}$
32: **end for**
33: **Output** w_a, w_b, and w_c

IV. EXPERIMENT

We evaluate the proposed method on two datasets, **CIFAR-10** and **ANIMAL-10N**.

Table 1: Information of the datasets in the experiment

	number of training data	number of testing data	Number of classes	image size
CIFAR10	50000	10000	10	32 × 32
ANIMAL10N	50000	5000	10	64 × 64

A. Datasets

1) **CIFAR-10 [7]**

CIFAR-10 is an initially clean dataset consisting of 60,000 images in 10 classes, with 6,000 images per class. The dataset is split into 50,000 images for training and 10,000 images for testing. Note that the classes are mutually exclusive.

Since we aim to train our model using noisy labels, we will manually corrupt the training set with three different levels of label noise, symmetric 40%, symmetric 80%, and asymmetric 40%.

At the end of the training, we will use the testing set with clean labels to evaluate our method.

2) **ANIMAL-10N [8]**

ANIMAL-10N is a real-world noisy dataset consisting of 55,000 images of 10 animal categories, which were crawled from online search engines and annotated by 15 participants. Since there are five pairs of confusing animals in the ten classes (i.e., wolves and coyotes), participants inevitably mislabeled the images with a noise rate of about 8%. The training dataset contains 50,000 images and the testing dataset contains 5,000 images.

B. Implementation Details

1) **Environment Setup**

The algorithm is implemented in Pytorch and all the experiments are carried out on Windows 10 with a Core i7 processor and NVIDIA GeForce GTX 1060 6GB.

2) **Baselines**

We compare our proposed full model (Algorithm 1), NFTT, with Co-teaching and Co-teaching with the Noise Filter. All methods are systematically compared in Table 2.

Table 2: Comparison with different methods

	Co-teaching	Co-teaching with Noise Filter	NFTT
Pre-trained	×	✓	✓
Network Structure	9-layers CNN	9-layers CNN	9-layers CNN

3) **Network Structure and Optimizer**

We adopt the 9-layers CNN model [2] as described in the Co-teaching paper for all methods listed above. The details of the architecture are in Table 3.

Adam optimizer with initial learning of 0.001 and momentum of 0.9 is adopted. For all experiments, in the stage of noise filtering (except for Co-teaching), we run 20 epochs for training the Noise Filter and the learning rate will start to decay at the 10-th epoch. In the stage of Co-teaching or Tri-teaching, we run 50 epochs and the corresponding learning rate will start to decay at the 30-th epoch.

Also, we will use cross-entropy loss to compute the loss of predicted labels against the ground-truth one.

Table 3: 9-layers CNN models

Deep CNN on CIFAR-10	Deep CNN on ANIMAL-10N
3×3 conv, 128 LReLU 3×3 conv, 128 LReLU 3×3 conv, 128 LReLU	
2×2 max-pool, stride 2 dropout, p = 0.25	
3×3 conv, 256 LReLU 3×3 conv, 256 LReLU 3×3 conv, 256 LReLU	
2×2 max-pool, stride 2 dropout, p = 0.25	
3×3 conv, 512 LReLU 3×3 conv, 256 LReLU 3×3 conv, 128 LReLU	
avg-pool	
dense 128→10	dense 128→10

4) **Other Setups**

In Algorithm 1, we set $R(T_t) = 1 - min\{\frac{T_t}{T_k}\tau, \tau\}$ where τ is set to be the noise rate and T_k is set to be 10 by following the strategy in Co-teaching Algorithm [2].

For measuring the performance, it is evaluated by using the test accuracy = $\frac{\#\ of\ correct\ predictions}{\#\ of\ testing\ data}$.

C. Experimental Results

1) **Results on CIFAR-10**

Table 4 compares our proposed method with Co-teaching and Co-teaching with the Noise Filter on CIFAR-10 of three different noise levels. For a fair comparison, all the experimental setups are the same.

For symmetric 40%, the performance of NFTT is almost the same as Co-teaching by outperforming 0.001%.

For symmetric 80%, our method achieves the best performance. NFTT outperforms Co-teaching by about 16.26% and Co-teaching with the Noise Filter by about 0.23%.

For asymmetric 40%, our method also achieves the best performance. NFTT outperforms Co-teaching by about 0.21% and Co-teaching with the Noise Filter by about 2.68%.

As shown in Table 4, the performance of NFTT and Co-teaching is essentially the same for symmetric 40% and asymmetric 40%, and the performance of NFTT is slightly better in comparison. However, when the noise rate is very high (symmetric 80%), NFTT will work much better than Co-teaching. The details of the model analysis will be discussed in the next subsection.

Table 4: Average test accuracy on CIFAR10 over the last five epochs

50 Epochs	Co-teaching	Co-teaching with Noise Filter	NFTT
Symmetric 40%	87.186%	86.59%	**87.187%**
Symmetric 80%	46.27%	62.30%	**62.53%**
Asymmetric 40%	85.69%	83.22%	**85.90%**

Note that we retested each task on the same computer, so the results are slightly different from the presentation slides. The results of all the experiments are available on GitHub (under {ROOT}/model_result).

2) **Results on ANIMAL-10N**

Table 5 compares our proposed method with Co-teaching and Co-teaching with the Noise Filter on ANIMAL-10N. The results demonstrate that NFTT also performs slightly better than Co-teaching on ANIMAL-10N by 0.06%.

Although the noise rate of ANIMAL-10N is only about 8%, it is natural that all methods perform worse on this dataset than on CIFAR-10 since ANIMAL-10N is a real-world noisy dataset containing five pairs of confusing animals. When tested on ANIMAL-10N, we adjusted the number of training epochs for the Noise Filter to 15 epochs, and the learning rate will start to decay at the 8th epoch.

Table 5: Average test accuracy on ANIMAL-10N over the last five epochs

	Co-teaching	Co-teaching with Noise Filter	NFTT
ANIMAL-10N	71.27%	69.08%	**71.33%**

D. Model Analysis

1) **Noise Filter**

Figure 3 illustrates the actual noise rate of the noisy CIFAR-10 training datasets before and after noise filtering. The actual noise rate of symmetric 40% (resp. symmetric 80% and asymmetric 40%) is 36% (resp. 71% and 20%). After applying the pre-trained Noise Filter, the noise rate of symmetric 40% (resp. symmetric 80% and asymmetric 40%) corrupted dataset is reduced to 2% (resp. 20% and 6%).

The reduced noise rate is considerably low; however, the amount of filtered training data may be insufficient for Tri-teaching. That is, for symmetric 80% (resp. symmetric 40% and asymmetric 40%), only 10,203 (resp. 28339 and 38538) data points are retained, and the pure ratio of the retained dataset is 80% (resp. 98% and 94%), where 8,141 (resp. 27737 and 36258) out of 10,203 (resp. 28339 and

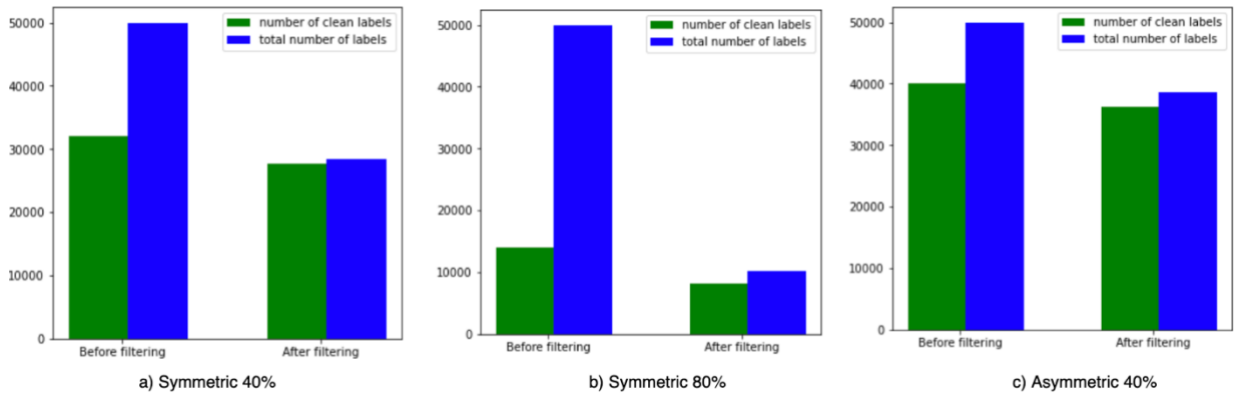

Figure 3. Performance of the Noise Filter

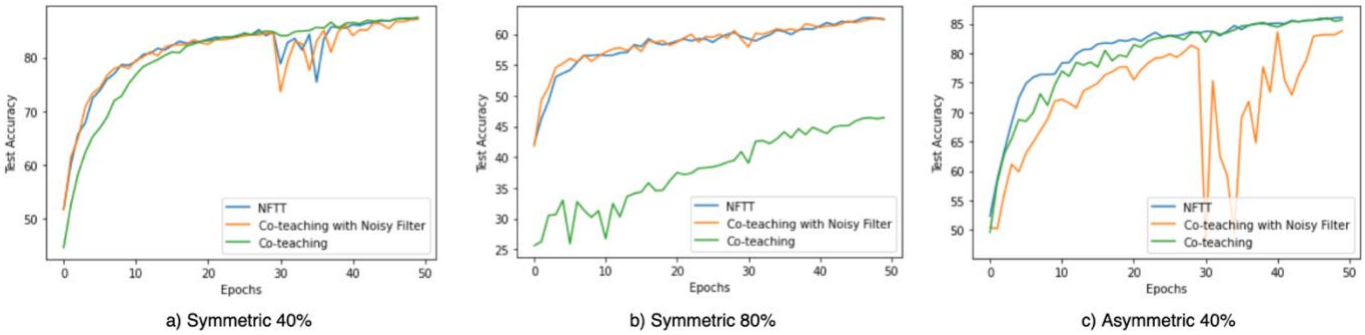

Figure 4. Test accuracy on CIFAR-10 over 50 epochs

38538) data points are clean. Insufficient retained training data may result in an ineffective performance improvement, especially in cases where the original noise rate is not high, such as symmetric 40% and asymmetric 40% tasks. One possible solution is to increase the number of epochs for training the Noise Filter to retain more training data for the second stage. However, it is a challenge to determine at which epoch we should stop training before the deep neural network starts to memorize the noisy labels.

2) **Performance Visualization**

Figure 4 shows the test accuracy of the different methods on three different noise levels of CIFAR-10 over 50 epochs.

Figures 4a and 4c intuitively illustrate that the performance of Co-teaching and NFTT is about the same for symmetric 40% and asymmetric 40%. However, our method could be quite effective in improving the performance of Co-teaching for the symmetric 80% task, namely, when the noise rate is remarkably high.

Since it requires significantly more time to train the model with NFTT than with Co-teaching, we suggest first measuring the original actual noise rate and applying Co-teaching when the noise rate is not high, otherwise applying NFTT.

3) **CNN Structure**

We also validate our method with CNNs of different architectures. The details of the architecture of a shallower CNN with 6-layers are shown in Table 6.

Table 6: 6-layers CNN models

Shallow CNN on CIFAR-10	Deep CNN on CIFAR-10
3×3 conv, 128 LReLU 3×3 conv, 128 LReLU	
2×2 max-pool, stride 2 dropout, p = 0.25	
3×3 conv, 256 LReLU	
2×2 max-pool, stride 2 dropout, p = 0.25	
3×3 conv, 512 LReLU 3×3 conv, 256 LReLU 3×3 conv, 128 LReLU	

avg-pool	
dense 128→10	dense 128→10

Tables 7 and 8 compare the results of NFTT with a deeper 9-layers CNN and a shallower 6-layers CNN on CIFAR-10 and ANIMAL-10N. The results indicate that NFTT with a 6-layer CNN will have better results if the noise rate is high, otherwise, NFTT with a 9-layer CNN will perform better.

Therefore, we suggest that after noise filtering, if the noise rate is still higher than 20%, NFTT with shallow CNN is applied, otherwise, NFTT with deep CNN will be a better choice.

Table 7: Average test accuracy of NFTT with different CNN structures on CIFAR10 over the last five epochs

50 Epochs	NFTT with deep CNN	NFTT with shallow CNN
Symmetric 40%	**87.187%**	86.14%
Symmetric 80%	62.53%	**62.97%**
Asymmetric 40%	**85.90%**	77.26%

Table 8: Average test accuracy of NFTT with different CNN structures on ANIMAL-10N over the last five epochs

50 Epochs	NFTT with deep CNN	NFTT with shallow CNN
ANIMAL-10N	**71.33%**	69.35%

V. Limitations and Future Work

A. Limitations

Due to hardware limitations, it takes an excessive amount of time to run a training epoch, which would make it infeasible for us to obtain more accurate test results over more training epochs.

B. Future Work

For the next step, we would try to improve the performance of our method by tuning the parameters such as the learning rate, the number of training epochs, forget rate, etc. In addition, the architecture of the CNN is another critical factor that affects performance.

In the current implementation of the noise filtering stage, we choose to remove the detected mislabeled samples directly from the noisy training dataset, which may result in insufficient training samples to improve performance, as described in Section IV. In future work, we will try label correction to ensure that there is enough training data rather than directly removing noisy data points.

One problem is that as the training epoch increases, the three networks will converge, resulting in Tri-teaching becoming self-teaching similar to MentorNet. [6] To resolve this problem, we can follow the strategy of "Updating by Disagreement", bridging with Tri-teaching, as described in [6], which can be considered as a Tri-teaching version of Co-teaching+. Therefore, the implementation of Tri-teaching+ with the Noise Filter is a promising attempt in the next step.

VI. Conclusion

In this report, we present our proposed method, NFTT, for improving the performance of image classification with noisy labels. In our method, a noise filter model is designed to reduce the noise rate of the training dataset by removing potentially noisy data points, and Tri-teaching is proposed to improve the classification performance. An extensive experiment on CIFAR-10 and ANIMAL-10N validates that NFTT can outperform Co-teaching, especially in the case of extremely high noise rates. In future work, we would like to explore an upgraded version of Tri-teaching that would improve the performance of the current method.

References

[1] Ding, Y., Wang, L., Fan, D., & Gong, B. (2018). A Semi-Supervised Two-Stage Approach to Learning from Noisy Labels.

[2] Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., & Sugiyama, M. (2018). *Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels*.

[3] Liu, S., Niles-Weed, J., Razavian, N., & Fernandez-Granda, C. (2020). *Early-Learning Regularization Prevents Memorization of Noisy Labels*.

[4] Jiang, L., Zhou, Z., Leung, T., Li, L.-J., & Fei-Fei, L. (2018). MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels.

[5] Malach, E., & Shalev-Shwartz, S. (2018). Decoupling “when to update” from “how to update.”

[6] Yu, X., Han, B., Yao, J., Niu, G., Tsang, I., & Sugiyama, M. (2019). *How does Disagreement Help Generalization against Label Corruption?*

[7] Krizhevsky, A. (2009b). Learning Multiple Layers of Features from Tiny Images.

[8] Song, H., Kim, M., and Lee, J., "SELFIE: Refurbishing Unclean Samples for Robust Deep Learning," In Proc. 36th Int'l Conf. on Machine Learning (ICML), Long Beach, California, June 2019