Robust Approaches for Learning with Noisy Labels

ROBUST APPROACHES FOR LEARNING WITH NOISY LABELS

BY

YANGDI LU, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of philosophy (2022) McMaster University

(Department of Computing and Software) Hamilton, Ontario, Canada

TITLE:	Robust Approaches for Learning with Noisy Labels
AUTHOR:	Yangdi Lu M.Sc., (Computing and Software) McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Wenbo He
NUMBER OF PAGES:	xxx, 189

To my wife, Eden, Y. Xu.

Thanks for love, support, understanding and encouragement.

Acknowledgements

I want to express my sincere gratitude to anyone who helped me during the years of my studies at McMaster University. First, my deepest gratitude goes to Prof. Wenbo He, my advisor, co-author, and friend, for her professional guidance, unwavering encouragement, and trust. Prof. He is well experienced in the areas of big data, security and trustworthy machine learning applications. Through frequent discussions with her, I have gained many valuable ideas. None of the work included here would have happened without her advice. Looking back on my graduate career, I am very fortunate to have Prof. He as my supervisor.

I would like to thank Prof. Ridha Khedri, and Prof. Reza Samavi for their insightful comments and suggestions on my thesis work during annual committee meetings. I am grateful to Prof. Minglun Gong for his kind help to serve as an external thesis appraiser. I would also like to thank Prof. Rong Zheng for serving as my defense chair. It was a privilege for me to have each of them serve on my doctoral committee.

In addition, I would like to thank all the members of my research group for their supports. Special thank goes to my colleagues and friends, Dr. Yang Bo and Dr. Shusheng Li. I have benefited a lot from their helps and stimulating discussions. It was a great pleasure for me to study and work with them.

Finally, my sincere thanks to all the members of my family, for their love, understanding, and support throughout my life. I share with them every piece of achievement that I have obtained.

Lay Abstract

Machine Learning has been highly successful in data-intensive applications but is often hampered when datasets contain noisy labels. Recently, Learning with Noisy Labels (LNL) is proposed to tackle this problem. By using techniques from LNL, the models can still generalize well even when trained on the data containing noisy supervised information. In this thesis, we study this crucial problem and provide a comprehensive analysis to reveal the core issue of LNL. We then propose five different methods to effectively reduce the learning errors in LNL. We show that our approaches achieve comparable or better performance compared to the state-of-the-art approaches on benchmark datasets with simulated label noise and real-world noisy datasets.

Abstract

Deep neural networks (DNNs) have achieved remarkable success in data-intense applications, while such success relies heavily on massive and carefully labeled data. In practice, obtaining large-scale datasets with correct labels is often expensive, time-consuming, and sometimes even impossible. Common approaches of constructing datasets involve some degree of error-prone processes, such as automatic labeling or crowdsourcing, which inherently introduce noisy labels. It has been observed that noisy labels severely degrade the generalization performance of classifiers, especially the overparameterized (deep) neural networks. Therefore, studying noisy labels and developing techniques for training accurate classifiers in the presence of noisy labels is of great practical significance. In this thesis, we conduct a thorough study to fully understand LNL and provide a comprehensive error decomposition to reveal the core issue of LNL. We then point out that the core issue in LNL is that the empirical risk minimizer is unreliable, i.e., the DNNs are prone to overfitting noisy labels during training. To reduce the learning errors, we propose five different methods, 1) Co-matching: a framework consists of two networks to prevent the model from memorizing noisy labels; 2) SELC: a simple method to progressively correct noisy labels and refine the model; 3) NAL: a regularization method that automatically distinguishes the mislabeled samples and prevents the model from memorizing them; 4)

EM-enhanced loss: a family of robust loss functions that not only mitigates the influence of noisy labels, but also avoids underfitting problem; 5) MixNN: a framework that trains the model with new synthetic samples to mitigate the impact of noisy labels. Our experimental results demonstrate that the proposed approaches achieve comparable or better performance than the state-of-the-art approaches on benchmark datasets with simulated label noise and large-scale datasets with real-world label noise.

Notation, Definitions, and Abbreviations

Abbreviations

AI	Artifical Intelligence
APL	Active and Passive Loss
CE	Cross Entropy
CNNs	Convolutional Neural Networks
DNNs	Deep Neural Networks
ELR	Early Learning Regularization
ELP	Early Learning Phenonmenon
EM	Entropy Minimization
ER	Entropy Regularization
FL	Focal Loss
FSL	Few-shot Learning

GCE	Generalized Cross Entropy
GMM	Gaussian Mixture Model
GNNs	Graph Neural Networks
KNN	K Nearest Neighbors
LNL	Learning with Noisy Labels
LS	Label Smoothing
MAE	Mean Absolute Error
NAL	Noise Attention Learning
NCE	Normalized Cross Entropy
NER	Negative Entropy Regularization
NLNL	Negative Learning with Noisy Labels
NLS	Negative Label Smoothing
PDF	Probability Density Function
PLC	Progressive Label Correction
PMD	Polynomial Margin Diminishing
RCE	Reversed Cross Entropy
ResNet	Residual Network
SELC	Self Ensemble Label Correction
SGD	Stochastic Gradient Descent
SL	Symmetric Loss
TCE	Taylor Cross Entropy

Publications and the Statement of Co-Authorship

Conference Papers:

[1] Lu, Yangdi, Bo, Yang and He, Wenbo. "An Ensemble Model for Combating Label Noise." In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM), pp. 608-617, doi: 10.1145/3488560.3498376.

Yangdi Lu researched the ensemble framework consists of multiple networks in the noisy labels scenario, proposed the algorithm, wrote the code, and composed the manuscript; Yang Bo reviewed the paper; Wenbo He provided the supervision of this work. The main contributor to this paper is Yangdi Lu (contributes more than 80%).

[2] Lu, Yangdi and He, Wenbo. "SELC: Self-Ensemble Label Correction Improves Learning with Noisy Labels." Published at International Joint Conference on Artificial Intelligence (IJCAI) 2022, Pages 3278-3284. https://doi.org/10.24963/ijcai.2022/455

Yangdi Lu researched the label correction methods in the noisy labels scenario, proposed the algorithm, wrote the code, and composed the manuscript; Wenbo He provided the supervision of this work. The main contributor to this paper is Yangdi Lu (contributes more than 80%).

[3] Lu, Yangdi, Bo, Yang and He, Wenbo. "Noise Attention Learning." Accepted to

NeurIPS 2022.

Yangdi Lu researched the regularization methods in the noisy labels scenario, proposed the algorithm, wrote the code, and composed the manuscript; Yang Bo reviewed the paper; Wenbo He provided the supervision of this work. The main contributor to this paper is Yangdi Lu (contributes more than 80%).

[4] Lu, Yangdi and He, Wenbo. "Entropy Minimization for Deep Learning with Noisy Labels." Plan to Submit to AAAI 2023.

Yangdi Lu researched the robust loss functions in the noisy labels scenario, proposed the algorithm, wrote the code, and composed the manuscript; Wenbo He provided the supervision of this work. The main contributor to this paper is Yangdi Lu (contributes more than 80%).

[5] Lu, Yangdi, and He, Wenbo. "MixNN: Combating Noisy Labels in Deep Learning by Mixing with Nearest Neighbors." In 2021 IEEE International Conference on Big Data (Big Data), pp. 847-856, doi: 10.1109/BigData52589.2021.9671816.

Yangdi Lu researched the augmentation algorithms in the noisy labels scenario, proposed the algorithm, wrote the code, and composed the manuscript; Wenbo He provided the supervision of this work. The main contributor to this paper is Yangdi Lu (contributes more than 80%).

Declaration of Academic Achievement

An Ensemble Model for Combating Label Noise

1. Proposed an framework fed with different augmented inputs to achieve the better ensemble effect, wherein an unsupervised matching loss is used to mitigate the influence of noisy labels and improve the generalization.

2. Proposed to fit a mixture model to per-loss distribution to estimate the clean probability of samples, which is used to select the high-confident clean samples for network parameter update.

3. Conducted extensive experiments on both simulated and real-world noisy datasets to demonstrate that our approach significantly advances state-of-the-art results.

SELC: Self-Ensemble Label Correction Improves Learning with Noisy Labels

1. Proposed a simple and effective label correction method SELC based on self-ensembling.

2. Proposed an effective metric based on unsupervised loss modeling to detect the turning point without requiring the test set and noise information.

3. Demonstrated that SELC achieves superior results on both simulated and real-world noisy datasets.

Noise Attention Learning

1. Proposed a simple yet effective method that automatically distinguishes the mislabeled samples and prevents the model from memorizing them.

2. Demonstrated that NAL achieves superior results on both simulated and real-world noisy datasets.

Entropy Minimization for Learning with Noisy Labels

1. Provided new insights into robust learning with noisy labels by reducing the model output fluctuations (or improving the output confidence).

2. Theoretically proved that by applying entropy minimization (EM) to non-robust loss functions, they can be robust to noisy labels without suffering from the underfitting problem under certain noise rate.

3. Demonstrated that the proposed loss functions outperform the state-of-the-art robust loss functions by a substantial margin.

MixNN: Combating Noisy Labels in Deep Learning by Mixing with Nearest Neighbors

1. Provide insights into representation distribution in early learning stage and proposed to generate new synthetic samples to robustly train DNNs, by aggregating the original samples with their top-K nearest neighbours.

2. Estimated the dynamic weights by unsupervised learning in the creation of synthetic samples. The weights are proportional to the clean probability of samples, thereby maintaining correct information while eliminating the wrong information from the mislabeled samples.

3. Demonstrated that the proposed method outperforms the state-of-the-art methods on benchmarks with simulated label noise and real-world noisy datasets.

Contents

List of Figures

Chapter 1

Introduction

"Can machines think?" This innovative question was raised in Alan Turing's paper entitled "Computing Machinery and Intelligence" (Turing, 2009). Suppose putting a machine player in an "imitation game", he stated that the best strategy for the machine is to try to provide answers that would naturally be given by a man. In other words, the ultimate goal of machines is to be as intelligent as humans. Over the past few decades, with the emergence of advanced models and algorithms (e.g. convolutional neural networks (CNNs) (Krizhevsky *et al.*, 2012), transformers (Vaswani *et al.*, 2017)), large-scale data sets (e.g. ImageNet (Deng *et al.*, 2009) with 1000 image classes), powerful computing frameworks and devices (e.g. GPU and distributed platforms), AI speeds up its pace to be like humans and supports many fields of daily life, such as search engines, autonomous driving cars, and industrial robots.

Albeit its prosperity, the superior performance of current deep neural networks (DNNs) owns much to the availability of large-scale correctly annotated datasets. For most of the supervised learning tasks, we always expect and assume a perfectly labeled large-scale training set. However, it is extremely time-consuming and expensive,

sometime even impossible to label a new dataset containing fully correct annotations. Typically, creating a regular dataset requires two steps: data collection and labeling process, involving two kinds of noise in the literature — feature noise and label noise (Zhu and Wu, 2004). Feature noise corresponds to the corruption of data features, while label noise refers to the change of label from its actual class (e.g. by incorrectly annotating a dog label to a cat image). Both noise types can cause a significant decrease in the performance, while label noise is considered to be more harmful (Frénay and Verleysen, 2013) as the label is unique for each sample while features are multiple. For example, video data contain audio, script and vision feature. The importance of each feature varies while the label always has a significant impact.

In typical labeling process, it is extremely expensive and time-consuming to label extensive data with high-quality annotations. To alleviate this problem, one may obtain the data with lower quality annotations efficiently through online keywords queries (Li *et al.*, 2017). Similarly, the expensive labeling process can be crowdsourced with the help of platforms such as Amazon Mechanical Turk [1] and Crowdflower [2], which effectively decrease labeling cost. Another widely used approach is to label data with automated systems. However, all these approaches inevitably introduce the label noise. Moreover, label noise can occur even in the case of expert annotators, such as brain images [3]. Even domain experts make mistakes because data can be extremely complex to be classified correctly (Lloyd *et al.*, 2004). Also, label noise can be injected intentionally to protect patients' privacy (van den Hout and van der Heijden, 2002).

A example of label noise is provided in Figure 1.1. When training the DNNs with

[1] http://www.mturk.com
[2] http://crowdflower.com
[3] https://adni.loni.usc.edu/

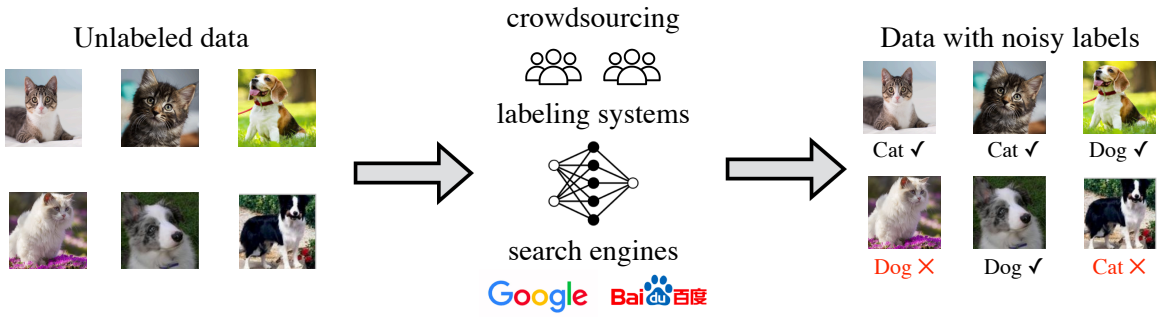


Figure 1.1: The generation of noisy labels.

noisy labels, a recent study (Zhang *et al.*, 2018a) shows that DNNs can easily overfit to noisy labels and results in poor generalization performance. To effectively learn from the data with noisy supervised information, a new machine learning paradigm called Learning with Noisy Labels (LNL) is proposed. LNL not only helps relieve the burden of labeling large-scale supervised data, but also enhances the robustness of models. Driven by the academic goal for AI to approach humans and the industrial demand for inexpensive learning, LNL has drawn much recent attention and is now a hot topic. Here we first define the problem of LNL and investigate the core issue of LNL through error decomposition.

1.1 Preliminaries

LNL is a sub-area in machine learning, before giving the formal definition of LNL, let us recall how machine learning is defined in the existing literature.

Definition 1.1.1 (Machine Learning (Mitchell *et al.*, 1997))**.** *A computer program is*

said to ***learn*** *from experience E with respect to some classes of task T and performance measure P, if its performance at tasks in T, as measured by P, can improve with E.*

The above definition can be generalized to a very wide range of practical applications. For example, consider an image classification task (T), a machine learning program improves its classification accuracy (P) through E obtained by training on a large number of labeled images (e.g. the ImageNet). Typically, existing machine learning applications, especially using deep neural networks as in the example mentioned above, require a lot of data samples with correct supervision information. However, this may be difficult or sometimes even impossible in real-world applications. LNL is a special and more general case of machine learning, which targets at obtaining good learning performance given noisy supervised information in the training set, which consists of examples of inputs $\boldsymbol{x}_i$'s along with their corresponding output $\bar{y}_i$'s. Formally, we define LNL in Definition 1.1.2.

Definition 1.1.2. ***Learning with Noisy Labels*** *(LNL) is a type of machine learning problems (specified by $\bar{E}$, T and P), where $\bar{E}$ is corrupted version of invisible clean E, consists of clean and mislabeled examples for the target task T.*

1.1.1 Core Issue of LNL

In any machine learning problem, usually there are prediction errors and one cannot obtain perfect predictions. In this section, we illustrate the core issue of LNL based on error decomposition in supervised machine learning (Bottou and Bousquet, 2007). This analysis applies to LNL including classification and regression tasks.

Notations

Consider a learning task T, LNL deals with a dataset $\bar{D} = \{\bar{D}_{\text{train}}, D_{\text{test}}\}$ consisting of a noisy training set $\bar{D}_{\text{train}} = \{(\boldsymbol{x}_i, \bar{y}_i)\}_i^N$, and a clean testing set D_{test}. Let $p(\boldsymbol{x}, y)$ be the ground truth joint probability distribution of input $\boldsymbol{x}$ and output y, $\bar{p}(\boldsymbol{x}, \bar{y})$ be the corrupted joint probability distribution of input $\boldsymbol{x}$ and output $\bar{y}$. Let $\hat{h}$ be the optimal hypothesis from $\boldsymbol{x}$ to y. LNL learns to discover $\hat{h}$ by fitting $\bar{D}_{\text{train}}$ and testing on D_{test}. For clarity, we assume $\bar{D}_{\text{c\_train}}$ be a set of clean training samples (i.e. inputs with correct labels) and $\bar{D}_{\text{m\_train}}$ be the mislabeled training samples (i.e. inputs with wrong labels). We have $\bar{D}_{\text{train}} = \bar{D}_{\text{c\_train}} \cup \bar{D}_{\text{m\_train}}$. Note that $\bar{D}_{\text{c\_train}}$ and $\bar{D}_{\text{m\_train}}$ are imagination sets which are unobservable. We define them only for clear explanations.

To approximate $\hat{h}$, the LNL model determines a hypothesis space $\mathcal{H}$ of hypotheses $h(\cdot;\theta)$'s, where θ denotes all the parameters used by h. A LNL algorithm is an optimization strategy that searches $\mathcal{H}$ to find the θ that parameterizes the best $h^* \in \mathcal{H}$. The LNL performance is measured by a loss function $\ell(\cdot,\cdot)$ defined over the prediction $h(\boldsymbol{x};\theta)$ and the observed output y over the test set.

Empirical Risk Minimization.

Given a hypothesis h, we want to minimize its *expected risk* R, which is the loss measured with respect to $p(\boldsymbol{x}, y)$. Specifically,

$$R(h) = \int \ell(h(\boldsymbol{x}), y) dp(\boldsymbol{x}, y) = \mathbb{E}[\ell(h(\boldsymbol{x}), y)]. \tag{1.1}$$

As $p(\boldsymbol{x}, y)$ is unknown, similar to regular machine learning tasks, the *empirical risk*, i.e., the average of sample losses over the noisy training set $\bar{D}_{\text{train}}$ of N samples,

$$\begin{aligned} R_N(h) &= \frac{1}{|\bar{D}_{\text{train}}|} \sum_{(\boldsymbol{x},\bar{y})\in\bar{D}_{\text{train}}} \ell(h(\boldsymbol{x}), \bar{y}) \\ &= \underbrace{\frac{1}{|\bar{D}_{\text{c\_train}}|} \sum_{(\boldsymbol{x},\bar{y})\in\bar{D}_{\text{c\_train}}} \ell(h(\boldsymbol{x}), \bar{y})}_{R_{N_{\text{c}}}(h)} + \underbrace{\frac{1}{|\bar{D}_{\text{m\_train}}|} \sum_{(\boldsymbol{x},\bar{y})\in\bar{D}_{\text{m\_train}}} \ell(h(\boldsymbol{x}), \bar{y})}_{R_{N_{\text{m}}}(h)} \end{aligned} \tag{1.2}$$

is usually used as a proxy for $R(h)$, leading to *empirical risk minimization* (Mohri *et al.*, 2018). However, in this case, minimizing $R_N(h)$ usually leads to an estimation of $\bar{p}(\boldsymbol{x}, \bar{y})$, which is completely different from $p(\boldsymbol{x}, y)$. Therefore, directly training models without any adjustment has been observed to lead to poor generalization performance (Zhang *et al.*, 2018a). Here we can decouple the $R_N(h)$ into $R_{N_{\text{c}}}(h)$ and $R_{N_{\text{m}}}(h)$. Since $\bar{D}_{\text{c\_train}}$ is a set of clean samples, finding a hypothesis that only minimizes $R_{N_{\text{c}}}(h)$ rather than $R_{N_{\text{m}}}(h)$ leads to a better estimation of $p(\boldsymbol{x}, y)$. For better illustration, let

- $\hat{h} = \arg\min_h R(h)$ be the function that minimizes the expected risk;
- $h^* = \arg\min_{h\in\mathcal{H}} R(h)$ be the function in $\mathcal{H}$ that minimizes the expected risk;
- $h_N = \arg\min_{h\in\mathcal{H}} R_N(h)$ be the function in $\mathcal{H}$ that minimizes the empirical risk;
- $h_{N_{\text{c}}} = \arg\min_{h\in\mathcal{H}} R_{N_{\text{c}}}(h)$ be the function in $\mathcal{H}$ that only minimizes the empirical risk of $\bar{D}_{\text{c\_train}}$ rather than $\bar{D}_{\text{m\_train}}$.

As $\hat{h}$ is unknown, one has to approximate it by searching some $h \in \mathcal{H}$. h^* is the best approximation for $\hat{h}$ in $\mathcal{H}$. h_N is the best hypothesis in $\mathcal{H}$ obtained by minimizing

the whole empirical risk $R_N(h)$, while h_{N_c} is the optimal hypothesis in $\mathcal{H}$ that only minimizes $R_{N_c}(h)$. For simplicity, we assume that $\hat{h}$, h^*, h_N, and h_{N_c} are unique. The *total error* can be decomposed as

$$\mathbb{E}[R(h_N) - R(\hat{h})] = \underbrace{\mathbb{E}[R(h^*) - R(\hat{h})]}_{\mathcal{E}_{\text{app}}(\mathcal{H})} + \underbrace{\mathbb{E}[R(h_{N_c}) - R(h^*)]}_{\mathcal{E}_{\text{est}}(\mathcal{H},N_c)} + \underbrace{\mathbb{E}[R(h_N) - R(h_{N_c})]}_{\mathcal{E}_{\text{fit}}(\mathcal{H},N,N_c)},$$

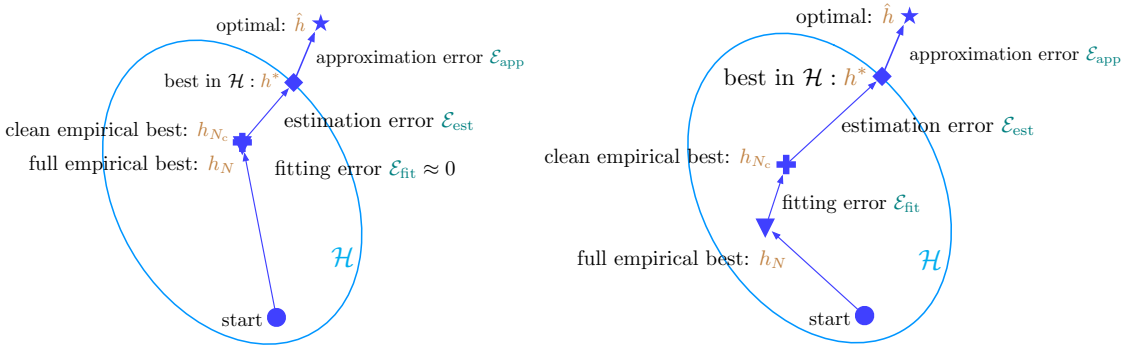
where the expectation is with respect to the random choice of $\bar{D}_{\text{train}}$. The *approximation error* $\mathcal{E}_{\text{app}}(\mathcal{H})$ measures how close the functions in $\mathcal{H}$ can approximate the optimal hypothesis $\hat{h}$. The *estimation error* $\mathcal{E}_{\text{est}}(\mathcal{H}, N_c)$ measures the effect of minimizing the clean empirical risk $R_{N_c}(h)$ instead of the expected risk $R(h)$ within $\mathcal{H}$. The *fitting error* $\mathcal{E}_{\text{fit}}(\mathcal{H}, N, N_c)$ measures the effect of minimizing the full empirical risk $R_N(h)$ instead of only the clean empirical risk $R_{N_c}(h)$.

As can be observed, the total error is influenced by $\mathcal{H}$ (hypothesis space), N (number of samples in $\bar{D}_{\text{train}}$) and N_c (number of samples in $\bar{D}_{\text{c\_train}}$). A special case is when $N = N_c$, the LNL reduces to regular learning problem.

Therefore, reducing the total error can be attempted from the perspectives of (1) data, which provides $\bar{D}_{\text{train}}$ and $\bar{D}_{\text{c\_train}}$; (2) model, which determines $\mathcal{H}$; and (3) algorithms, which searches for the optimal hypothesis h_{N_c} that only fits $\bar{D}_{\text{c\_train}}$.

Unreliable Empirical Risk Minimizer.

In LNL, the model would easily fit all noisy samples. The empirical risk $R_N(h)$ may then be far from being a good approximation of the expected risk $R(h)$, and the resultant empirical risk minimizer h_{N} overfits. Indeed, this is the core issue of LNL, i.e., the empirical risk minimizer h_{N} is no longer reliable. Therefore, LNL is much



(a) Learning with clean data (b) Learning with noisy data

Figure 1.2: Comparison of learning with clean and noisy data.

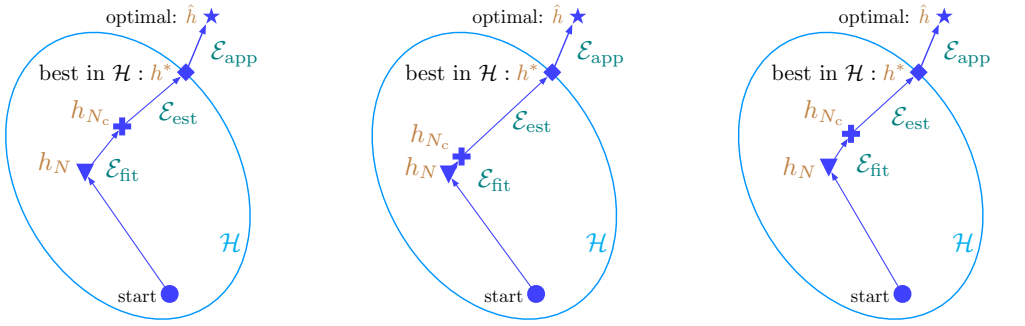
harder. A comparison of learning with clean data and noisy data is shown in Figure 1.2. Compared to learning with clean data, both the estimation error and fitting error of LNL increase.

1.1.2 Related Work

Almost all the existing works aim to reduce the learning errors to achieve noise robustness, we summarize them into three different categories (see Figure 1.3).

Reduce estimation error $\mathcal{E}_{\text{est}}$

Similar to learning with clean data, the estimation error can be reduced by increasing the number of samples. Therefore, some methods use prior knowledge to augment $\bar{D}_{\text{train}}$. For example, Mixup (Zhang *et al.*, 2018b) constructs virtual training samples by linearly combining two random samples' features and labels. Nishi *et al.* (2021) evaluated multiple augmentation strategies and found that using one set of augmentations for loss modeling tasks and another set for learning is the most effective in



(a) Reduce $\mathcal{E}_{\text{est}}$ (b) Reduce $\mathcal{E}_{\text{fit}}$ (c) Reduce both $\mathcal{E}_{\text{est}}$ and $\mathcal{E}_{\text{fit}}$

Figure 1.3: Different perspectives on how existing methods solve the LNL problem.

LNL. Our method MixNN (Lu and He, 2021) dynamically mixes the sample with its nearest neighbors to generate synthetic samples for noise robustness. Other methods leverage the unlabeled data to improve the performance of LNL. For example, Garg *et al.* (2021) augmented the training data with random labeled data and provided a theoretical analysis that ensures the true risk is lower. Iscen *et al.* (2022) utilized the unlabeled data to enforce the consistency of model predictions, resulting in improving the performance.

Reduce fitting error $\mathcal{E}_{\text{fit}}$

Some methods aim to prevent the model from overfitting to mislabeled samples. 1) *Regularization*: these methods implicitly restrict the model parameters or adjusts the gradients to prevent the model from memorizing mislabeled samples. For example, Li *et al.* (2020b) proved the gradient descent with early stopping is an effective regularization to achieve robustness to label noise. Hu *et al.* (2019) added the regularizer to limit the distance between the model parameters to initialization for noise robustness. ELR (Liu *et al.*, 2020) estimates the target by temporal ensembling (Laine and Aila,

2016) and adds a regularization term to cross-entropy loss to avoid memorization of mislabeled samples. Our method NAL scales the gradients according to the cleanliness of different samples to achieve noise robustness. 2) *Robust loss functions*: these methods develop loss functions that are inherently robust to label noise, including DMI (Xu *et al.*, 2019), MAE (Ghosh *et al.*, 2017), GCE (Zhang and Sabuncu, 2018), SCE (Wang *et al.*, 2019), NCE (Ma *et al.*, 2020), TCE (Feng *et al.*, 2021) and GJS (Englesson and Azizpour, 2021). We propose to increase the output confidence by adding EM term to ensure noise robustness for any loss. 3) *Sample selection*: The key idea is trying to select clean samples or reweigh the samples in training. During the early learning stage, the samples with smaller loss values are more likely to be the clean samples. Based on this observation, MentorNet (Jiang *et al.*, 2018) pre-trains a mentor network for assigning weights to samples for guiding the training of the student network. Decoupling (Malach and Shalev-Shwartz, 2017) updates the two networks by using the samples having different predictions. Co-teaching (Han *et al.*, 2018) trains two networks which select small-loss samples within each mini-batch to train each other. Co-teaching+ (Yu *et al.*, 2019) improves it by updating the network on disagreement data to keep the two networks diverged. Ren *et al.* (2018) reweighed samples based on their gradient directions. JoCoR (Wei *et al.*, 2020) jointly trains two networks with the examples that have prediction agreement between two networks. We propose a novel framework (Co-matching) with two networks fed with different strengths of augmented inputs to achieve the better ensemble effect. 4) *Loss correction*: These methods correct the loss by estimating the noise transition matrix. Patrini *et al.* (2017) estimated the label corruption matrix for loss correction. Hendrycks *et al.* (2018) improved the corruption matrix by using a clean set of data.

Reduce both $\mathcal{E}_{\text{est}}$ and $\mathcal{E}_{\text{fit}}$

Some methods focus on correcting the noisy labels, so that the model is gradually refined. Reed *et al.* (2015) proposed a bootstrapping method which modifies the loss with model predictions. Ma *et al.* (2018) improved the bootstrapping method by exploiting the dimensionality of feature subspaces to dynamically reweigh the samples. Arazo *et al.* (2019) improved bootstrapping using a dynamic weighting scheme through unsupervised learning techniques. PLC (Zhang *et al.*, 2020) progressively corrects the labels when the prediction confidence over a dynamic threshold. Our method SELC gradually correct noisy labels by ensemble predictions.

1.1.3 Noise Assumptions

It is extremely challenging to design robust methods for the real-world noisy datasets directly as the noise pattern in these datasets are unpredictable. To simplify the problem, noise assumptions are usually pre-defined to simulate the generation of label noise. Generally, there are two noise assumptions: class-conditional label noise and instance-dependent label noise.

Class-conditional Label Noise

The noise flip probability of class-conditional label noise depends only on labels, not on the input features (Natarajan *et al.*, 2013). The true label is corrupted by a noise transition matrix Q, where $Q_{ij} = \Pr[\bar{y} = j|y = i]$ given that noisy label $\bar{y}$ is flipped from clean label y. This matrix has two representative models *symmetric* and *asymmetric*. Symmetric noise is generated by uniformly flipping the label to one of the other class label. Asymmetric flipping is a simulation of fine-grained classification

$$Q = \begin{bmatrix} 1-\epsilon & \frac{\epsilon}{1-\epsilon} & \dots & \frac{\epsilon}{1-\epsilon} & \frac{\epsilon}{1-\epsilon} \\ \frac{\epsilon}{1-\epsilon} & 1-\epsilon & \frac{\epsilon}{1-\epsilon} & \dots & \frac{\epsilon}{1-\epsilon} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \frac{\epsilon}{1-\epsilon} & \dots & \frac{\epsilon}{1-\epsilon} & 1-\epsilon & \frac{\epsilon}{1-\epsilon} \\ \frac{\epsilon}{1-\epsilon} & \dots & \frac{\epsilon}{1-\epsilon} & \frac{\epsilon}{1-\epsilon} & 1-\epsilon \end{bmatrix} \quad Q = \begin{bmatrix} 1-\epsilon & \epsilon & 0 & \dots & 0 \\ 0 & 1-\epsilon & \epsilon & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & 1-\epsilon & \epsilon \\ \epsilon & 0 & \dots & 0 & 1-\epsilon \end{bmatrix}$$

(a) Q for symmetric noise (b) Q for asymmetric noise

Figure 1.4: Noise transition matrix for symmetric and asymmetric label noise.

with noisy labels in the real world, where the mistakes only occur within very similar classes. Since the noise only depends on the labels, the flip probability of samples in the same class is identical. Figure 1.4 shows the noise transition matrix of symmetric and asymmetric label noise when noise rate is ϵ.

Instance-dependent Label Noise

Instance-dependent label noise assumes that the noise flip probability is dependent on different instances, which is more complex than class-conditional label noise as different samples have different flip probabilities. For example, Xia *et al.* (2020) proposed the partial instance-dependent label noise on face image data, where the noise is based on different parts (the contribution of each part, such as nose, eye and mouth, is different). Zhang *et al.* (2020) proposed polynomial margin diminishing (PMD) noise which allows arbitrary noise level except for data far away from the true decision boundary. This is consistent with the real-world scenario, i.e., data near the decision boundary are harder to distinguish and more likely to be mislabeled, while a datum far away from the decision boundary is a typical example of its true class. Chen *et al.* (2021a) proposed a method to generate the instance-dependent label noise by using the second highest confident prediction, since the labels are most likely to

Dataset	# Train	# Val	# Test	# Classes	Input size	Noise rate (%)	Year	Source
Benchmarks with clean labels								
MNIST	60K	N/A	10K	10	28×28	≈ 0.0	1998	NIST
Fashion MNIST	60K	N/A	10K	10	28×28	≈ 0.0	2017	Zalando
SVHN	73K	N/A	26K	10	32×32	≈ 0.0	2011	Google Street View
CIFAR-10	50K	-	10K	10	32×32	≈ 0.0	2009	80 Million Tiny Images
CIFAR-100	50K	-	10K	100	32×32	≈ 0.0	2009	80 Million Tiny Images
Tiny-ImageNet	100K	10K	10K	200	64×64	≈ 0.0	2015	ImageNet
Real-world datasets with unpredictable label noise								
ANIMAL-10N	50K	N/A	5K	10	64×64	≈ 8	2019	crowdsourcing
Food-101N	310K	5K	25K	101	384×384	≈ 18.4	2018	web search
Clothing1M	1M	14K	10K	14	224×224	≈ 38.5	2015	shopping websites
Webvision	2.4M	50K	50K	1000	256×256	≈ 20.0	2017	web search

Table 1.1: Summary of widely-used datasets for learning with noisy labels.

flip to other class with highest confidence.

1.1.4 Dataset

Base on these noise assumptions (or models), we can manually corrupt the clean benchmarks to obtain simulated noisy datasets. The common methodology in this area is first designing robust algorithms under certain noise assumption. Then we evaluate the algorithms on the real-world datasets to see their effectiveness. We summarize the datasets widely-used in this research area in Table 1.1, and introduce the details of each dataset as follows.

MNIST

The Modified National Institute of Standards and Technology (MNIST) dataset (LeCun, 1998) consists of handwritten digits images. It has a training set of 60,000 examples, and a test set of 10,000 examples. Each example is a 28×28 grayscale image, associated with a label from 10 classes (i.e. from 0 to 9). For asymmetric

label noise, $2 \rightarrow 7$, $3 \rightarrow 8$, $5 \leftrightarrow 6$ and $7 \rightarrow 2$.

Fashion MNIST

The Fashin MNIST dataset (Xiao *et al.*, 2017) is a dataset of Zalando's article images. It also consists of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes (e.g. T-shirt, Dress). For asymmetric noise, $boot \rightarrow sneaker$, $sneaker \rightarrow sandals$, $pullover \rightarrow shirt$ and $coat \leftrightarrow dress$.

SVHN

The Street View Housing Numbers (SVHN) dataset (Netzer *et al.*, 2011) consists of images depicting house numbers, which range from 0 to 9. Each example is a 32×32 color image. It has a training set of 73,257 examples, and a test set of 26,032 examples. The way to generate asymmetric noise is similar to MNIST.

CIFAR-10 and CIFAR-100

The CIFAR-10 dataset (Krizhevsky *et al.*, 2009) consists of natural color images, each of size 32×32 pixels. Each image is classified into 1 of 10 classes, such as dog, cat, or ship. The training set contains 50,000 images, while the test set contains 10,000 image. The CIFAR-100 dataset (Krizhevsky *et al.*, 2009) is similar to CIFAR-10, except it has 100 classes containing 600 images each. The 100 classes are grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs). For CIFAR-10, the asymmetric noise is generated by flipping $truck \rightarrow automobile$, $bird \rightarrow airplane$,

$deer \rightarrow horse$ and $cat \leftrightarrow dog$. For CIFAR-100, the asymmetric noise flips each class into the next, circularly within super-classes.

Tiny-ImageNet

The Tiny-ImageNet (Le and Yang, 2015) is a subset of the ImageNet dataset in the famous ImageNet Large Scale Visual Recognition Challenge (ILSVRC). It contains 100000 images of 200 classes (500 for each class) downsized to 64×64 colored images. Each class has 500 training images, 50 validation images and 50 test images.

ANIMAL-10N

The ANIMAL-10N dataset (Song *et al.*, 2019) contains 5 pairs of confusing animals with a total of 55,000 images (50,000 for training and 5,000 for test). The 5 pairs are (cat, lynx), (jaguar, cheetah), (wolf, coyote), (chimpanzee, orangutan), (hamster, guinea pig). The raw images are crawled from online search engines including Bing and Google using the above keywords. Then the images are assigned to 15 recruited participants and each of them is required to annotate 6,000 images within a week. Therefore, the noise rate of the dataset is around 8%.

Food-101N

The Food-101N dataset (Lee *et al.*, 2018) consists of about 310,009 crawled food images annotated by their search keywords in the Food-101 (Bossard *et al.*, 2014) taxonomy. The noise rate of the dataset is around 18.4%. It directly adopts the testing set of Food-101 to evaluate image classification.

Clothing1M

The Clothing1M dataset (Xiao *et al.*, 2015) contains 1 million images of clothing obtained from online shopping websites with 14 classes, such as T-shirt, sweater, and so on. The labels are generated by using the surrounding texts of the images that are provided by the sellers and thus contain many wrong labels. The label noise rate is around 38.46%, with some pairs of classes frequently confused with each other (e.g. knitwear and sweater). This dataset also contains 50k, 14k, and 10k of clean data for training, validation, and testing, respectively.

Webvision

The Webvision dataset (Li *et al.*, 2017) is a large web images dataset that contains more than 2.4 millions of images crawled from the Flickr and Google Images Search by using the 1000 concepts of ImageNet (Deng *et al.*, 2009). The label noise level of Webvision is estimated at 20%. Following (Chen *et al.*, 2019), we select the first 50 classes of Google image subset (around 66,000 images) for training. The number of test images is around 2,500.

1.2 Objective and Contributions

In LNL, the main objective is to design algorithms to learn the accurate DNNs even the training data containing noisy labels. The main contribution of this thesis shall be the investigation of the different techniques to achieve noise robustness on image classification task. We focus on sample selection, label correction, regularization, robust loss functions and augmentation strategy to improve performance. More specifically,

in this work we made the following contributions:

1. We proposed a framework, Co-matching, which consists of two networks for robust learning with noisy labels. In Co-matching, we used weak (e.g. using only crop-and-flip) and strong (e.g. RandAugment) augmentations for two networks respectively to ensure their divergence. The stronger augmented one produces disparate prediction compared to the weak one, which guarantees the better ensemble effect. Aside from the supervised classification loss, we introduced an unsupervised matching loss to keep the consistency of predictions from two networks without using noisy labels. We also utilized a mixture model to dynamically select the confident clean samples without the need to estimate noise rates. We conducted extensive experiments on both simulated and real-world noisy datasets. Experiments show that Co-matching significantly advances state-of-the-art results.

2. We proposed self-ensemble label correction (SELC), which potentially corrects noisy labels during training thus preventing the model from being affected by the noisy labels. SELC leverages the knowledge provided in the model predictions over historical training epochs to form a consensus of prediction (ensemble prediction). We demonstrated that combining ensemble prediction with the original noisy label leads to a better target. Accordingly, the model is gradually refined as the targets become less noisy, resulting in improving performance. We proposed a metric to estimate the turning point only using training data, allowing us to select a suitable initial epoch to perform SELC. Our experimental results shows that SELC achieves superior results and can be integrated with other techniques such as mixup (Zhang *et al.*, 2018b) to further enhance the performance.

3. We proposed Noise Attention Learning (NAL) which enjoys simplicity and

universality from a new perspective. The core idea behind NAL is to let the model automatically differentiate the mislabeled samples from clean samples, and prevent the memorization of mislabeled samples in training. Specifically, we introduced an attention branch that outputs scalar attention weights to indicate the predictive power of representations. We incorporated the attention weights in our loss function, so that the attention branch can be learned to divide samples, i.e., clean samples are learned to be associated with large weights, while mislabeled ones with small weights. Theoretically, we demonstrated that NAL neutralizes the effect of the mislabeled samples on the gradient and ensures the contribution of clean samples to the whole gradient remains dominant, thereby preventing the deep neural networks from overfitting mislabeled samples. Empirically, we demonstrated that the NAL achieves better robustness than the state-of-the-art methods on five noisy datasets.

4. We provided new insights into robust learning with noisy labels by reducing the model output fluctuations. Specifically, we proposed to minimize the entropy of prediction to enhance learning with label noise. We demonstrated that by applying entropy minimization to non-robust loss functions, they can be robust to noisy labels without suffering from the underfitting problem. We empirically demonstrated that the proposed loss functions outperform the state-of-the-art robust loss functions by a substantial margin.

5. We proposed MixNN that generates new synthetic samples to robustly train DNNs by aggregating the original samples with their top-K nearest neighbors. We estimated the dynamic weights by unsupervised learning in the creation of synthetic samples. The weights are proportional to the clean probability of samples, thereby maintaining correct information while eliminating the wrong information from the

mislabeled samples. We gradually estimated the soft labels in the loss function, yields enhancing the performance in the presence of extreme label noise. We demonstrated that the proposed method outperforms the state-of-the-art methods on two standard benchmarks with simulated label noise and two real-world noisy datasets.

1.3 Thesis Outline

This thesis is organized as follows. In Chapter 2, we introduce Co-matching, an ensemble framework consisting of two networks to achieve robust learning with noisy labels. In Chapter 3, we introduce SELC, a simple yet effective method to progressively correct the noisy labels and refine the model. In Chapter 4, we introduce a regularization method that scales the gradients to prevent the model from overfitting to noisy labels. In Chapter 5, we study the existing loss functions and propose to apply EM on any loss functions to achieve noise robustness. In chapter 6, we propose MixNN, an augmentation method for generating synthetic samples to mitigate the effect of noisy labels. We conclude the thesis in chapter 7.

Chapter 2

An Ensemble Model for Combating Label Noise

2.1 Citation and Main Contributor

Lu, Yangdi, Bo, Yang and He, Wenbo. "An Ensemble Model for Combating Label Noise." In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM), pp. 608-617, doi: 10.1145/3488560.3498376.
The main contributor to this paper is the first author - Yangdi Lu (contributes more than 80%).

2.2 Copyright

2.3 Introduction

Deep Neural Networks (DNNs) have become the par excellence approach to deal with a variety of computer vision tasks (Krizhevsky *et al.*, 2012; Liu *et al.*, 2019). However, the superior performance comes with the cost of requiring a large-scale training dataset with high-quality annotations. It is difficult to attain such strong supervision information due to the high cost of the manually labeling process. Hence, we turn to web services to obtain large-scale training data with labels, such as querying online search engines (Li *et al.*, 2017), collecting online websites images with surrounding texts (Xiao *et al.*, 2015), downloading social media images with tags (Mahajan *et al.*, 2018) or crowdsourcing (Yu *et al.*, 2018). However, these approaches usually inevitably introduce label noise. For example, Xiao *et al.* (2015) collected 1 million training images from online shopping websites and generates their labels from surrounding texts with 38.5% estimated label noise. Previous study (Zhang *et al.*, 2018a) demonstrates DNN can memorize noisy labels easily and generalize poorly on clean test data. Therefore, mitigating the effects of noisy labels has attracted considerable attention especially when the training data are from web resource.

To handle noisy labels, most approaches focus on estimating the noise transition matrix (Goldberger and Ben-Reuven, 2017; Patrini *et al.*, 2017; Xia *et al.*, 2019) and correcting the label according to model prediction (Reed *et al.*, 2015; Ma *et al.*, 2018; Tanaka *et al.*, 2018; Yi and Wu, 2019). Another promising direction of study is based on sample selection, which trains two networks simultaneously by using small-loss instances (Han *et al.*, 2018; Chen *et al.*, 2019; Yu *et al.*, 2019; Wei *et al.*, 2020). For instance, Decoupling (Malach and Shalev-Shwartz, 2017) and Co-teaching+ (Yu *et al.*, 2019) introduce the "Disagreement" strategy to keep the two networks diverged

to achieve better ensemble effects. However, the samples selected by "Disagreement" strategy are not guaranteed to have correct labels (Wei *et al.*, 2020), resulting in only a small portion of clean samples being utilized in the training process. Co-teaching (Han *et al.*, 2018) and JoCoR (Wei *et al.*, 2020) aim to reduce the divergence between two different networks so that the number of clean labels utilized in each mini-batch increases. In the beginning, two networks with different learning abilities filter out different types of error. However, with the increasing training epoch number, two networks gradually converge to a consensus and even make the wrong predictions consistently. Besides, these methods rely on a known noise rate to accurately select the small-loss samples in each mini-batch, which is usually impractical.

To address the above concerns, it is crucial to keep the balance between divergence and consistency of two networks through the whole training procedure. In this paper, we propose an ensemble model for robust learning with noisy labels. Specifically, we use weak (e.g. using only crop-and-flip) and strong (e.g. using RandAugment (Cubuk *et al.*, 2020)) augmentations for two networks respectively to avoid the consensus of their predictions. The stronger augmentation results in disparate prediction compared to the weak one, which guarantees the better ensemble effect due to the divergence between two networks. Aside from the supervised classification loss, we propose an unsupervised matching loss to keep the consistency of predictions from two networks without using noisy labels. According to the observation of loss distribution in early-learning stage, we propose a method based on a mixture model to dynamically select the confident clean samples without the need to estimate noise rates. In this way, the influence of mislabeled samples in the process of model learning is mitigated, and our framework is inherently noise-tolerant. The main contributions are summarized

as follows:

- We propose a framework with two networks fed with different augmented inputs to achieve the better ensemble effect, wherein an unsupervised matching loss is used to mitigate the influence of noisy labels and improve the generalization.
- We analyze the gradient of our loss function and show that the gradient term derived from unsupervised matching loss corrects the gradient of the cross-entropy loss.
- Different from the existing approaches that require estimating noise rates, we fit a mixture model to loss distribution to select the high-confident clean samples for network parameter update, making our method applicable in real life.
- We conduct extensive experiments on both simulated and real-world noisy datasets. Experiments show that our approach significantly advances state-of-the-art results. We study the effect of data augmentations and provide an ablation study to examine the influence of different components.

2.4 Related Work

Numerous methods have been proposed for robust classification with noisy labels. Herein, we briefly review the relevant existing approaches.

Curriculum learning. Inspired from human cognition, Curriculum learning (CL) (Bengio *et al.*, 2009) proposes to start from easy samples and go through harder samples to improve convergence and generalization. In the noisy label scenario, easy/hard

concepts are associated with clean/noisy samples. Based on CL, Ren *et al.* (2018) leveraged an additional validation set to adaptively assign weights to noisy samples for less loss contribution in every iteration.

Sample selection. Another set of emerging methods aim to select the clean labels out of the noisy ones to guide the training. Previous work (Arpit *et al.*, 2017) empirically demonstrates the *early-learning* phenomenon that DNNs tend to learn clean labels before memorizing noisy labels during training, which justifies that instances with small-loss values are more likely to be clean instances. Based on this observation, Lyu and Tsang (2019) proposed a curriculum loss that chooses samples with small-loss values for loss calculation. MentorNet (Jiang *et al.*, 2018) pre-trains a mentor network for selecting small-loss instances to guide the training of the student network. Nevertheless, similar to Self-learning approach, MentorNet inherits the same inferiority of accumulated error caused by the sample-selection bias.

Two classifiers with Disagreement and Agreement. Inspired by Co-training (Blum and Mitchell, 1998), Co-teaching (Han *et al.*, 2018) symmetrically trains two networks by selecting small-loss instances in a mini-batch for updating the parameters. These two networks could filter different types of errors brought by noisy labels since they have different learning abilities. When the error from noisy data flows into the peer network, it will attenuate this error due to its robustness (Han *et al.*, 2018). However, two networks converge to a consensus gradually with the increase of epochs. To tackle this issue, Co-teaching+ (Yu *et al.*, 2019) introduces the "Update by Disagreement" strategy (Malach and Shalev-Shwartz, 2017) which conducts updates only on selected instances, where there is a prediction disagreement between

two classifiers. Through this, the decision of "when to update" depends on a disagreement between two networks instead of depending on the noisy labels. As a result, it would reduce the dependency on noisy labels as well as keep two networks divergent. However, as noisy labels are spread across the whole space of examples, there may be very few clean labels in the disagreement area. Thus, JoCoR (Wei *et al.*, 2020) suggests jointly training two networks with the instances that have prediction agreement between two networks. However, the two networks in JoCoR are also prone to converge to a consensus and even make the same wrong predictions when datasets are under high noise ratio.

Other methods. Some approaches focus on creating noise-tolerant loss functions (Ghosh *et al.*, 2017; Zhang and Sabuncu, 2018; Wang *et al.*, 2019; Ma *et al.*, 2020). Other methods attempt to correct the loss (Reed *et al.*, 2015; Patrini *et al.*, 2017; Hendrycks *et al.*, 2018; Ma *et al.*, 2018; Tanaka *et al.*, 2018; Wang *et al.*, 2018; Arazo *et al.*, 2019; Song *et al.*, 2019). Many approaches (Ding *et al.*, 2018; Li *et al.*, 2020a) have been proposed to combat noisy labels through semi-supervised learning. These approaches (Hu *et al.*, 2019; Liu *et al.*, 2020; Lu *et al.*, 2021) introduce the regularization term to avoid memorization of noisy labels.

Our proposed approach is related to sample selection using two networks. However, it is also fundamentally different from existing methods. Instead of only using classification loss, we propose an extra matching loss to reduce the effect of noisy labels. We feed different augmented images to two networks respectively, yielding a stronger generalization ability. Beyond heuristic design, we provide the gradient analysis to explain how our loss function avoids memorization of the mislabeled samples. Different from selecting small-loss samples by using a fixed ratio, we use a mixture

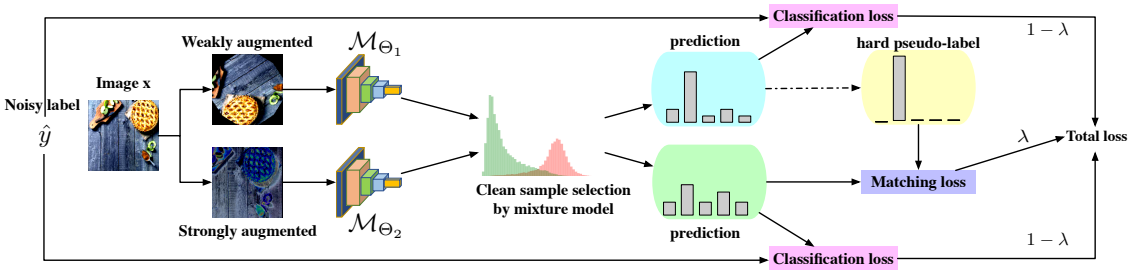


Figure 2.1: Our approach trains two networks ($\mathcal{M}_{\Theta_1}$ and $\mathcal{M}_{\Theta_2}$) simultaneously. A weakly-augmented version of an image $\boldsymbol{x}$ (top) is fed into the model $\mathcal{M}_{\Theta_1}$ to obtain its prediction (blue box). We convert the prediction to a one-hot hard pseudo-label as an anchoring label (yellow box). Then, we compute second model's prediction (green box) for a strongly-augmented version of the same image (bottom). The models are trained on total loss (i.e. the linear convex combination of the unsupervised matching loss and classification loss) to make the prediction on the strongly-augmented version match the anchoring label. For parameter update, we fit the Gaussian Mixture Model (GMM) to loss distribution for distinguishing the clean samples, which ensuring error caused by noisy labels would not be accumulated.

model to dynamically select the high-confident clean samples.

2.5 Methodology

2.5.1 Background

Our work aims to develop an algorithm to learn a classifier that achieves robust performance on the test set even the provided training data contains noisy labels. Consider the C-class classification problem in noisy label scenario, we have a training set $D = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ is an input and $\hat{\boldsymbol{y}}_i \in \{0,1\}^C$ is one-hot vector corresponding to $\boldsymbol{x}_i$. In the noisy label scenario, true label $\boldsymbol{y}_i$ is not observable in practice. The classification model maps each input $\boldsymbol{x}_i$ to a C-dimensional logits using a DNN

model $\mathcal{M}_\Theta$ and then feeds the logits into a softmax function to produce $\boldsymbol{p}_i$ of the conditional probability of each class.

$$\boldsymbol{p}_i = \texttt{softmax}(\mathcal{M}_\Theta(\boldsymbol{x}_i)) = \frac{e^{\mathcal{M}_\Theta(\boldsymbol{x}_i)}}{\sum_{c=1}^{C} e^{(\mathcal{M}_\Theta(\boldsymbol{x}_i))_c}}. \tag{2.1}$$

Θ denotes the parameters of the DNN and $(\mathcal{M}_\Theta(\boldsymbol{x}_i))_c$ denotes the c-th entry of logits $\mathcal{M}_\Theta(\boldsymbol{x}_i)$. Then the DNN is trained via the cross-entropy loss to measure how the model fits the training set D.

$$\ell_{ce}(D, \Theta) = -\frac{1}{N}\sum_{i=1}^{N} \hat{\boldsymbol{y}}_i^T \log(\boldsymbol{p}_i). \tag{2.2}$$

However, as noisy label $\hat{\boldsymbol{y}}_i$ is likely to be wrong, the model gradually memorizes the training samples with wrong labels when optimizing ℓ_{ce}. Existing studies (Zhang *et al.*, 2018a) have observed that DNNs fully overfit to noisy labels during training, causing the classification performance degradation. In addition, Liu *et al.* (2020) have observed and also theoretically proved that when trained on noisy labels, DNNs first fit the training data with clean labels during an *early learning* phase, before eventually *memorizing* the training data with wrong labels. This *early learning* phenomenon also reflects on the loss distribution which motivates us to develop an approach to select clean samples in Section 2.5.4.

2.5.2 Our approach: Co-matching

A diagram of our approach is shown in Figure 2.1. We name our approach *Co-matching* as it **Co**-trains two deep networks by minimizing the loss contains an unsupervised **matching** loss term. We denote the two deep networks in our model as

$\mathcal{M}_{\Theta_1}$ and $\mathcal{M}_{\Theta_2}$ with parameters Θ_1 and Θ_2 respectively. Thus $\texttt{softmax}(\mathcal{M}_{\Theta_1}(\boldsymbol{x}_i))$ and $\texttt{softmax}(\mathcal{M}_{\Theta_2}(\boldsymbol{x}_i))$ are the softmax probabilities for input $\boldsymbol{x}_i$ produced by $\mathcal{M}_{\Theta_1}$ and $\mathcal{M}_{\Theta_2}$. For the model inputs, we perform two types of augmentation for each network: weak and strong, denoted by $\alpha(\cdot)$ and $\mathcal{A}(\cdot)$ respectively. We will describe the forms of augmentation used for $\mathcal{A}(\cdot)$ and $\alpha(\cdot)$ in section 2.5.5. For notation simplicity, we denote $\boldsymbol{p}_i^{\Theta_1}$ and $\boldsymbol{p}_i^{\Theta_2}$ as abbreviations for predictions, i.e., $\texttt{softmax}(\mathcal{M}_{\Theta_1}(\alpha(\boldsymbol{x}_i)))$ and $\texttt{softmax}(\mathcal{M}_{\Theta_2}(\mathcal{A}(\boldsymbol{x}_i)))$ respectively. As the DNN model can easily overfit the noisy labels when trained with standard cross-entropy loss, resulting in poor classification performance. Our basic idea is to reduce the dependence of loss function on noisy labels. Therefore, the loss function in Co-matching exclusively consists of two loss terms: a supervised loss ℓ_c for classification task and an unsupervised matching loss ℓ_m for augmentation anchoring (i.e. encourage the two models to output the consistent predictions on different augmented data). So our total loss on dataset D is calculated as follows:

$$\mathcal{L}(D, \Theta_1, \Theta_2) = (1 - \lambda)\ell_c(D, \Theta_1, \Theta_2) + \lambda\ell_m(\boldsymbol{x}, \Theta_1, \Theta_2), \tag{2.3}$$

where $\lambda \in [0, 1]$ is a fixed scalar hyperparameter controlling the importance weight of the two loss terms. Since there are correctly-labeled samples remaining in noisy training data, our classification loss ℓ_c is the traditional cross-entropy loss over two

models.

$$\begin{aligned}\ell_c(D,\Theta_1,\Theta_2) &= \ell_{ce}(D,\Theta_1)+\ell_{ce}(D,\Theta_2)\\ &= -\frac{1}{N}\sum_{i=1}^{N}\hat{\boldsymbol{y}}_i^T\log(\boldsymbol{p}_i^{\Theta_1})-\frac{1}{N}\sum_{i=1}^{N}\hat{\boldsymbol{y}}_i^T\log(\boldsymbol{p}_i^{\Theta_2})\\ &= -\frac{1}{N}\sum_{i=1}^{N}\hat{\boldsymbol{y}}_i^T\log(\boldsymbol{p}_i^{\Theta_1}\odot\boldsymbol{p}_i^{\Theta_2}).\end{aligned} \tag{2.4}$$

Compared to cross-entropy loss in Eq. (2.2), our classification loss term ℓ_c is more resistant to noisy labels. On the one hand, assume $\hat{\boldsymbol{y}}_i$ is a correct label for $\boldsymbol{x}_i$, when minimizing ℓ_c, both $\boldsymbol{p}_i^{\Theta_1}$ and $\boldsymbol{p}_i^{\Theta_2}$ are updated toward $\hat{\boldsymbol{y}}_i$. To gain the optimal result of Hadamard product between $\boldsymbol{p}_i^{\Theta_1}$ and $\boldsymbol{p}_i^{\Theta_2}$, two models are required to produce more confident and consistent predictions close to $\hat{\boldsymbol{y}}_i$. On the other hand, assume $\hat{\boldsymbol{y}}_i$ is a wrong label for $\boldsymbol{x}_i$. Weakly augmented input can result in prediction $\boldsymbol{p}_i^{\Theta_1}$ close to $\hat{\boldsymbol{y}}_i$, while strongly augmented input generates disparate prediction $\boldsymbol{p}_i^{\Theta_2}$ compare to the weak one. It makes the result of Hadamard product between $\boldsymbol{p}_i^{\Theta_1}$ and $\boldsymbol{p}_i^{\Theta_2}$ more difficult to reach the wrong label $\hat{\boldsymbol{y}}_i$, resulting in mitigating the effect of overfitting noisy labels in both networks. Besides, it also explains that Co-matching can always keep two networks diverged throughout the whole training to achieve better ensemble effects. Nevertheless, solely keeping the divergence of two networks may not promote the learning ability to select clean samples, which is the main drawback of Co-teaching+ (Yu *et al.*, 2019). In addition, under high level of label noise (i.e. most of the nois labels $\hat{\boldsymbol{y}}_i$ are wrong), it is difficult for the supervised loss ℓ_c to learn a robust classifier. This motivates us to develop an extra loss term that does not require using noisy label $\hat{\boldsymbol{y}}_i$ but still improves the generalization ability.

Our basic idea is to use the model's prediction for a weakly augmented input

as the target label for the strongly augmented version of the same image. This maximizes the consistency of the two networks resulting in helping the model find a wider minimum and provides better generalization performance. In Co-matching, we compute an *anchor* (or anchoring label) for each sample by the prediction of model $\mathcal{M}_{\Theta_1}$. To obtain an anchoring label of given image $\boldsymbol{x}_i$, we get the predicted class distribution from model $\mathcal{M}_{\Theta_1}$ given a weakly-augmented version of the image: $\boldsymbol{p}_i^{\Theta_1}$ and $\boldsymbol{p}_i^{\Theta_1} = [(p_i^{\Theta_1})_1, (p_i^{\Theta_1})_2, \cdots, (p_i^{\Theta_1})_C]$. Then, we use hard pseudo-labeling way to get $\boldsymbol{t}_i^{\Theta_1}$ as the anchoring label.

$$(t_i^{\Theta_1})_j = \begin{cases} 1 & \text{if } j = \arg\max_c (p_i^{\Theta_1})_c \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

The use of hard pseudo-labeling has the similar function to entropy maximization (Grandvalet and Bengio, 2005), where the model's predictions are encouraged to be low-entropy (i.e., high-confidence). Besides, the hard pseudo-labeling is likely to reduce the negative effect of knowledge distillation caused by noisy labels in early learning stage. The anchoring label is used as the target probability for the prediction of a strongly-augmented version of same image in $\mathcal{M}_{\Theta_2}$. However, not all samples are suitable to generate anchoring labels, as hard pseudo-label of low-confident noisy samples may introduce unstable inconsistency. Thus, we develop an approach to select high-confident clean samples in Section 2.5.4.

We use the standard cross-entropy loss rather than mean squared error or Jensen-Shannon Divergence as it maintains stability and simplifies implementation. Thus,

the unsupervised matching loss is

$$\ell_m(\boldsymbol{x}, \Theta_1, \Theta_2) = -\frac{1}{N}\sum_{i=1}^{N} \boldsymbol{t}_i^{\Theta_1} \log(\boldsymbol{p}_i^{\Theta_2}). \tag{2.6}$$

By minimizing the total loss in Eq. (2.3), the model consistently improves the generalization performance under different levels of label noise. Under low-level label noise (i.e., 20%), supervised classification loss $\ell_c(D, \Theta_1, \Theta_2)$ takes the lead. Co-matching tends to learn from the most correctly-labeled samples. Under high-level label noise (i.e., 80%), unsupervised matching loss $\ell_m(\boldsymbol{x}, \Theta_1, \Theta_2)$ takes the lead such that Co-matching inclines to maximize the consistency of the networks to improves the generalization without requiring noisy labels.

2.5.3 Theoretical Analysis on Loss Function

We explain how our loss function can effectively prevent the model memorizing the mislabeled samples by analyzing the gradient. We first explain how the standard cross-entropy loss in Eq. (3.1) memorizes samples with wrong labels. The gradient of cross-entropy loss with respect to Θ equals

$$\nabla\ell_{ce}(D, \Theta) = \frac{1}{N}\sum_{i=1}^{N} \nabla\mathcal{M}_{\Theta}(\boldsymbol{x}_i)\big(\boldsymbol{p}_i - \hat{\boldsymbol{y}}_i\big), \tag{2.7}$$

where $\nabla\mathcal{M}_{\Theta}(\boldsymbol{x}_i)$ is the Jacobian matrix of the DNN logits for the i-th input with respect to Θ. In clean training data scenario, $\boldsymbol{p}_i - \hat{\boldsymbol{y}}_i$ of true class entry will always be negative and rest entries are positive. Therefore, performing stochastic gradient descent increases the probability of true class and reduces the residual probabilities at other entries. However, in noisy-label scenario, if c is the true class, but c-th entry

of noisy label $(\hat{y}_i)_c = 0$, then the contribution of the i-th sample to $\nabla \ell_{ce}(D, \Theta)$ is reversed (i.e. $(p_i - \hat{y}_i)_c$ should be negative but get positive instead). In the meanwhile, the entry corresponding to the impostor class c', is also reversed because $(\hat{y}_i)_{c'} = 1$. Therefore, performing stochastic gradient descent eventually results in memorization of mislabeled samples.

However, our loss function can counteract this influence. Similarly, we derive the gradient of simplified $\mathcal{L}(D, \Theta_1, \Theta_2)$ ($\lambda = 0.5$) with respect to Θ_1 and Θ_2 equals

$$\begin{aligned}\nabla \mathcal{L}(D, \Theta_1, \Theta_2) &= \frac{1}{N} \sum_{i=1}^{N} \nabla \mathcal{M}_{\Theta_1, \Theta_2}(\boldsymbol{x}_i) \Big(\frac{\boldsymbol{p}_i^{\Theta_1} + \boldsymbol{p}_i^{\Theta_2}}{2} - \hat{\boldsymbol{y}}_i + \frac{\boldsymbol{p}_i^{\Theta_2} - \boldsymbol{t}_i^{\Theta_1}}{2} \Big) \\ &= \frac{1}{N} \sum_{i=1}^{N} \nabla \mathcal{M}_{\Theta_1, \Theta_2}(\boldsymbol{x}_i) \Big(\boldsymbol{p}_i^{\Theta_2} - \hat{\boldsymbol{y}}_i + \frac{\boldsymbol{p}_i^{\Theta_1} - \boldsymbol{t}_i^{\Theta_1}}{2} \Big) \end{aligned} \tag{2.8}$$

where $\nabla \mathcal{M}_{\Theta_1, \Theta_2}(\boldsymbol{x}_i)$ is the Jacobian matrix of the DNNs logits for the i-th input with respect to Θ_1 and Θ_2. If c is the true class, since $\boldsymbol{t}_i^{\Theta_1}$ is calculated by hard pseudo-labeling in Eq. (2.5), then the c-th entry of $\boldsymbol{p}_i^{\Theta_1} - \boldsymbol{t}_i^{\Theta_1}$ is negative. Compared to the gradient of cross-entropy loss, we have an additional negative term $(\boldsymbol{p}_i^{\Theta_1} - \boldsymbol{t}_i^{\Theta_1})/2$ to adjust the gradient coefficients, which is useful both for correctly-labeled and mislabeled samples. For correctly-labeled samples, the first term $\boldsymbol{p}_i^{\Theta_2} - \hat{\boldsymbol{y}}_i$ vanishes after the early-learning stage, allowing the mislabeled samples to dominate the gradient. Adding the negative term $(\boldsymbol{p}_i^{\Theta_1} - \boldsymbol{t}_i^{\Theta_1})/2$ counteracts this effect by ensuring that the magnitudes of the coefficients on correctly-labeled samples remains large. For mislabeled samples, the c-th entry of first term $(\boldsymbol{p}_i^{\Theta_2} - \hat{\boldsymbol{y}}_i)_c$ is positive because $(\hat{\boldsymbol{y}}_i)_c = 0$. Adding the negative term $((\boldsymbol{p}_i^{\Theta_1} - \boldsymbol{t}_i^{\Theta_1})/2)_c$ dampens the coefficients on these mislabeled samples, thereby diminishing their effect on the gradient. Thus, our loss function boosts the gradient of correctly-labeled samples and neutralizes the gradient

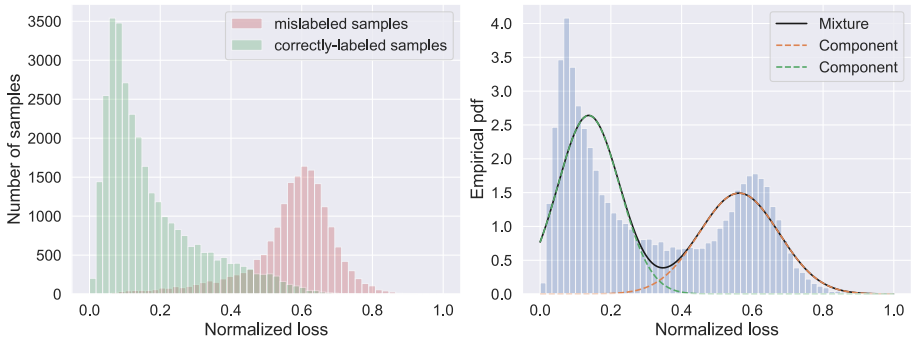


Figure 2.2: Train on CIFAR-10 with 40% label noise after 10 epochs with cross-entropy loss. Left: The ground truth normalized loss distribution. Right: The pdf of mixture model and two components after fitting a two component GMM to loss distribution.

of mislabeled samples, which prevents the second model memorizing the noisy labels.

2.5.4 Clean Sample Selection

Since DNNs learn clean patterns before memorizing noisy labels (Arpit *et al.*, 2017), small-loss instances are more likely to be the ones that are correctly labeled (Han *et al.*, 2018). Training the model only using correctly-labeled instances in each mini-batch data would be resistant to noisy labels. This approach is named as "small-loss" trick and is widely used in existing works (Han *et al.*, 2018; Yu *et al.*, 2019; Wei *et al.*, 2020). However, it requires a given noise rate or estimating the noise rate using additional steps to decide how many samples are selected. In this paper, we observe that the correctly-labeled samples can be distinguished from the loss distribution alone. To estimate the probability of being correctly-labeled sample, we introduce a two component Gaussian Mixture Model (GMM) (Permuter *et al.*, 2006) to fit the normalized loss distribution as shown in Figure 2.2. The probability density function

(pdf) of GMM with K components on the loss ℓ can be defined as

$$P(\ell) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\ell \mid \mu_k, \sigma_k^2), \quad \sum_{k=1}^{K} \pi_k = 1, \tag{2.9}$$

where π_k are the mixing coefficient for the linear convex combination of each individual pdf $\mathcal{N}(\ell \mid \mu_k, \sigma_k^2)$. In our case, we use an Expectation-Maximization (EM) algorithm to estimate the π_k, μ_k and σ_k^2. Therefore, we can obtain the probability of a sample being correctly-labeled or mislabeled through the posterior probability:

$$P(k \mid \ell) = \frac{P(k)P(\ell \mid k)}{P(\ell)} = \frac{\pi_k \mathcal{N}(\ell \mid \mu_k, \sigma_k^2)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\ell \mid \mu_k, \sigma_k^2)} \tag{2.10}$$

where $k = 1(2)$ indicate correct (wrong) labels. Note that we always calculate the cross-entropy loss to estimate the clean probability for all samples after every epoch. But we use our loss defined in Eq. (2.3) for training the model which contains other loss term to deal with label noise. Then we select the clean samples in n-th mini-batch D_n for updating the network parameters as follows

$$\hat{D}_n = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) \mid P(k = 0 \mid \ell_{ce}(\boldsymbol{x}_i)) > 0.5 \text{ and } (\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) \in D_n\}. \tag{2.11}$$

After obtaining the confident clean samples set $\hat{D}_n$ in mini-batch n, we calculate the loss for these examples to further conduct back propagation.

$$\mathcal{L}(\hat{D}_n, \Theta_1, \Theta_2) = -\frac{1}{|\hat{D}_n|} \sum_{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) \in \hat{D}_n} \hat{\boldsymbol{y}}_i^T \log(\boldsymbol{p}_i^{\Theta_1} \odot \boldsymbol{p}_i^{\Theta_2}) + \boldsymbol{t}_i^{\Theta_1} \log(\boldsymbol{p}_i^{\Theta_2}). \tag{2.12}$$

Therefore, our approach effectively select the clean samples without a need to estimate the noise rate. Put all these together, our algorithm is described in Algorithm 1. We

Algorithm 1: Co-matching

Input: two networks $\mathcal{M}_{\Theta_1}$ and $\mathcal{M}_{\Theta_2}$ with parameters $\Theta = \{\Theta_1, \Theta_2\}$, weak augmentation $\alpha(\cdot)$, strong augmentation $\mathcal{A}(\cdot)$, importance weight λ, training set D, batch size B, learning rate η, number of training epochs T;

$\Theta_1, \Theta_2 =$ Warmup(D, Θ_1, Θ_1); `// train with cross-entropy loss 10 epochs for warmup.`

for $t = 1, 2, \ldots, T$ **do**

  $P(k = 0 \mid \ell_{ce}(\boldsymbol{x}_i)) =$ GMM $(D, \ell_{ce}, \Theta_1, \Theta_2)$; `// model loss distibution to obtain clean probability for each sample.`

  Shuffle D into $\frac{|D|}{B}$ mini-batches ;

  for $n = 1, 2, \ldots, \frac{|D|}{B}$ **do**

    Fetch n-th mini-batch D_n from D ;

    Calculate the prediction $\boldsymbol{p}^{\Theta_1} = \texttt{softmax}(\mathcal{M}_{\Theta_1}(\alpha(\boldsymbol{x}))), \forall \boldsymbol{x} \in D_n$;

    Calculate the prediction $\boldsymbol{p}^{\Theta_2} = \texttt{softmax}(\mathcal{M}_{\Theta_2}(\mathcal{A}(\boldsymbol{x}))), \forall \boldsymbol{x} \in D_n$;

    Calculate the anchoring label $\boldsymbol{t}^{\Theta_1}$ by Eq. (2.5) ;

    Obtain confident clean samples by Eq. (2.11) ;

    Calculate the loss by Eq. (2.12) ;

    Update $\Theta = \Theta - \eta \nabla \mathcal{L}(\hat{D}_n, \Theta_1, \Theta_2)$;

Output Θ_1 and Θ_2.

also compare Co-matching to other existing approaches in Appendix 2.8.3.

2.5.5 Augmentations in Co-matching

Our framework leverages two kinds of augmentations: "weak" and "strong". In our experiments, weak augmentation is a standard crop-and-flip augmentation strategy. Specifically, we randomly crop the images and flip them horizontally with a probability of 50% on all datasets. As for "strong" augmentation, we adopt RandAugment (Cubuk *et al.*, 2020), which is based on AutoAugment (Cubuk *et al.*, 2019). AutoAugment learns an augmentation strategy based on transformations from the Python Imaging Libraries [1] using reinforcement learning. Given a collection of transformations (e.g., color inversion, contrast adjustment, translation, etc.), RandAugment

[1] https://www.pythonware.com/products/pil/

randomly selects M transformations for each sample in a mini-batch. As originally proposed, RandAugment uses a single fixed global magnitude that controls the severity of all distortions (Cubuk *et al.*, 2020). Instead of optimizing the hyperparameter magnitude by using grid search, we find that sampling a random magnitude from a pre-defined range at each training step (instead of using a fixed global value) works better for learning with noisy labels. The implementation details are in Section 2.6.3.

2.6 Experiments

2.6.1 Experimental Settings

We evaluate our method on two benchmarks with simulated label noise, CIFAR-10 and CIFAR-100 (Krizhevsky *et al.*, 2009), and one real-world dataset, Clothing1M (Xiao *et al.*, 2015). Clothing1M consists of 1 million training images collected from online shopping websites with noisy labels generated from surrounding texts. CIFAR-10 and CIFAR-100 are initially clean. Following (Patrini *et al.*, 2017), we corrupt the datasets by label transition matrix Q, where $Q_{ij} = \Pr[\hat{y} = j|y = i]$ given that noisy label $\hat{y}$ is flipped from clean label y. The matrix Q has two representative label noise models: (1) Symmetric flipping (Van Rooyen *et al.*, 2015) is generated by uniformly flipping the label to one of the other class label; (2) Asymmetric flipping (Patrini *et al.*, 2017) is a simulation of fine-grained classification with noisy labels in the real world, where the mistakes only occur within very similar classes. More details are described in Appendix 2.8.1.

Networks and optimizer. For CIFAR-10 and CIFAR-100, we use a 7-layer network architecture for fair comparison with (Wei *et al.*, 2020). The Adam optimizer

			Same Category Methods				Other Category Methods			
	Noise ratio/Method	Standard	Decoupling	Co-teaching	Co-teaching+	JoCoR	GCE	SL	APL	Co-Matching (Ours)
CIFAR-10	Symmetric-20%	69.57 ± 0.20	69.55 ± 0.20	78.07 ± 0.24	78.66 ± 0.20	85.69 ± 0.06	89.93 ± 0.08	89.13 ± 0.16	85.54 ± 0.51	**90.07 ± 0.16**
	Symmetric-50%	42.48 ± 0.35	41.44 ± 0.46	71.54 ± 0.17	57.13 ± 0.46	79.32 ± 0.37	81.38 ± 0.06	79.76 ± 0.20	80.66 ± 0.13	**87.63 ± 0.22**
	Symmetric-80%	15.79 ± 0.37	15.64 ± 0.42	27.71 ± 4.39	24.13 ± 5.54	25.97 ± 3.11	44.33 ± 0.15	53.62 ± 0.37	44.19 ± 4.40	**58.86 ± 1.54**
	Asymmetric-40%	69.36 ± 0.23	69.46 ± 0.08	73.75 ± 0.34	69.03 ± 0.30	76.38 ± 0.32	74.17 ± 0.45	74.39 ± 0.75	78.37 ± 0.02	**82.32 ± 0.57**
CIFAR-100	Symmetric-20%	35.46 ± 0.25	33.21 ± 0.22	43.71 ± 0.20	49.15 ± 0.24	52.43 ± 0.20	57.83 ± 0.73	47.60 ± 0.34	59.92 ± 0.48	**60.47 ± 0.29**
	Symmetric-50%	16.87 ± 0.13	15.03 ± 0.33	34.30 ± 0.39	39.08 ± 0.73	42.73 ± 0.96	49.24 ± 0.30	31.66 ± 1.43	52.26 ± 0.37	**53.81 ± 0.56**
	Symmetric-80%	4.08 ± 0.21	3.80 ± 0.01	14.95 ± 0.15	15.00 ± 0.42	14.41 ± 0.60	32.18 ± 0.27	13.51 ± 0.82	26.28 ± 2.05	**35.23 ± 0.82**
	Asymmetric-40%	27.23 ± 0.45	26.25 ± 0.27	28.27 ± 0.22	30.45 ± 0.15	31.52 ± 0.31	40.02 ± 0.35	37.25 ± 0.16	**42.25 ± 0.30**	39.14 ± 0.37

Table 2.1: Average test accuracy (%) on CIFAR-10 and CIFAR-100 over the last 10 epochs. The results (mean ± std) are reported over 5 random runs and best results are in bold.

(momentum=0.9) is used with an initial learning rate of 0.001, and the batch size is set to 128. We run 200 epochs in total and linearly decay learning rate to zero from 80 to 200 epochs. As for Clothing1M, we accept ResNet18 (He *et al.*, 2016a) with ImageNet pretrained weights and use Adam optimizer (momentum=0.9) with a batch size of 64. We run 20 epochs in total and set learning rate to 8×10^{-4}, 5×10^{-4} and 5×10^{-5} for 5, 5 and 10 epochs respectively.

Metrics. To measure the performance, we use the test accuracy, i.e., *test accuracy* = (*# of correct predictions*) / (*# of test dataset*). Higher test accuracy means that the algorithm is more robust to the label noise. Following the (Han *et al.*, 2018; Wei *et al.*, 2020), we also calculate the label precision in each mini-batch, i.e., *label precision* = (*# of clean labels*) / (*# of all selected labels*). Specifically, we attain the clean set by GMM in each mini-batch, and then calculate the ratio of clean samples in the clean set. Intuitively, an algorithm with higher label precision is also more robust to the label noise (Han *et al.*, 2018; Wei *et al.*, 2020). However, we find that the higher label precision is not necessarily lead to higher test accuracy with extreme label noise in Co-matching, we will explore this phenomenon in Section 2.6.2.

Baselines. We compare Co-matching with four close-related methods, including

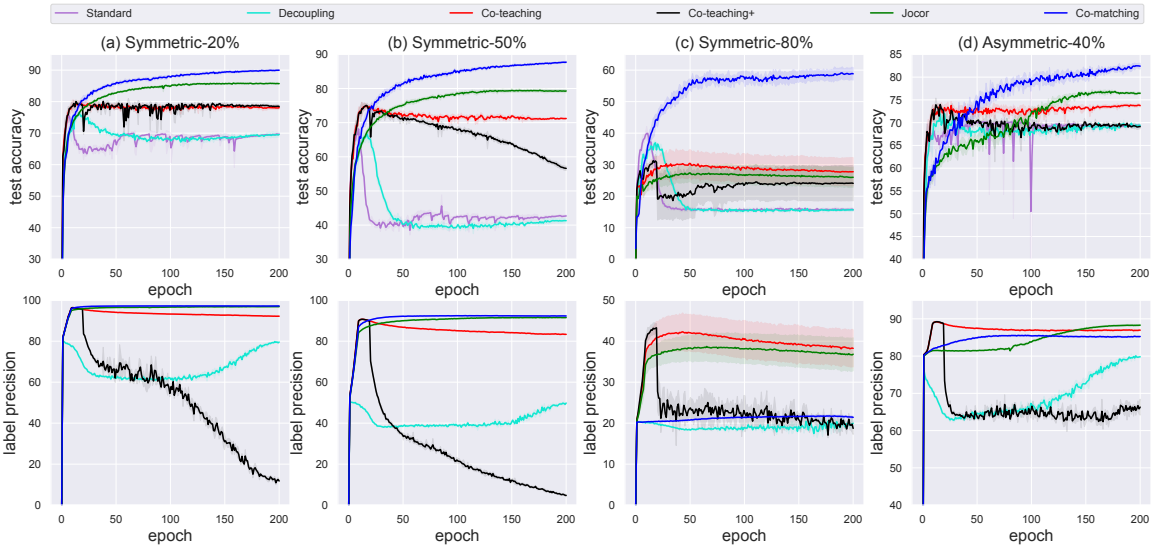


Figure 2.3: Results on CIFAR-10 dataset. Top: test accuracy(%) vs. epochs; bottom: label precision(%) vs. epochs.

Decoupling (Malach and Shalev-Shwartz, 2017), Co-teaching (Han *et al.*, 2018), Co-teaching+ (Yu *et al.*, 2019), JoCoR (Wei *et al.*, 2020) and three methods from different category, including GCE (Zhang and Sabuncu, 2018), SL (Wang *et al.*, 2019) and APL (Ma *et al.*, 2020). We implement all methods with same environment and default parameters by Pytorch. Note that all compared algorithms do not use extra techniques such as mixup (Zhang *et al.*, 2018b) to improve the performance. All results are reported over five random runs. The error bar for standard deviation in each figure has been highlighted as shade.

2.6.2 Comparison with the State-of-the-Arts

Results on simulated datasets CIFAR-10 and CIFAR-100 We report the average test accuracy over the last 10 epochs of all methods in Table 2.1. Co-matching

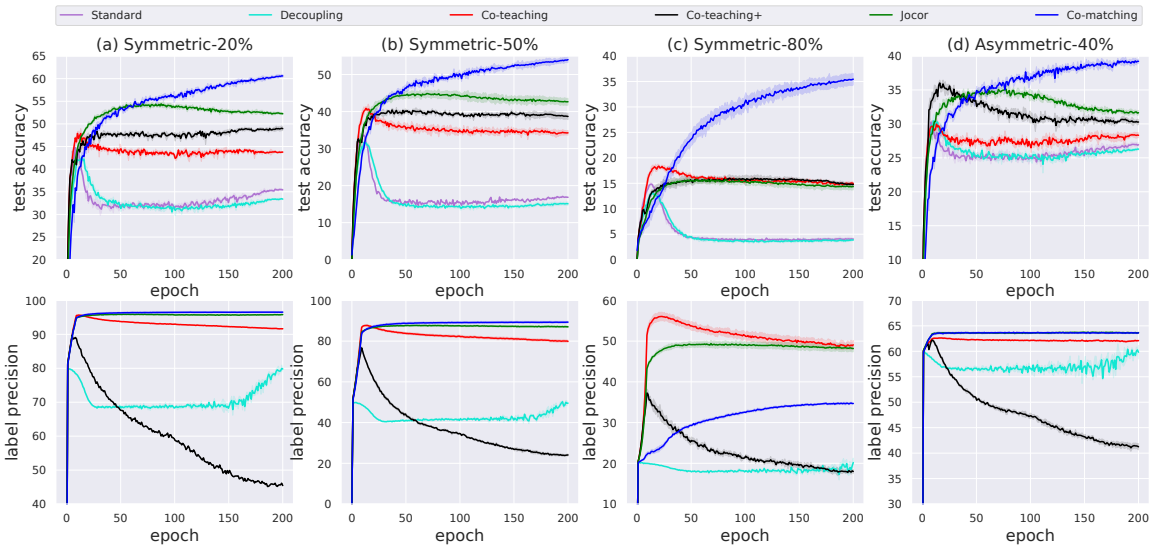


Figure 2.4: Results on CIFAR-100 dataset. Top: test accuracy(%) vs. epochs; bottom: label precision(%) vs. epochs.

outperforms other methods by a large margin for almost all noise rates across all datasets. We also find that Co-matching is more effective when the noise rates are extremely high. For instance, on CIFAR-10 with Symmetric-80% label noise, Co-matching outperform the best baseline method by more than 5.24%. Within the methods from same category, JoCoR performs better than other baselines in all noise cases except the hardest Symmetric-80% case. It means JoCoR has reduced to Co-teaching in function and suffers the same problem which the two networks converge to a wrong consensus, resulting in making the wrong predictions consistently. Note that APL sometimes delivers a relatively good performance, as it has guaranteed robustness to asymmetric label noise.

The top rows of Figure 2.3 and Figure 2.4 show the test accuracy vs. epochs of

close-related methods on CIFAR-10 and CIFAR-100. With different levels of symmetric and asymmetric label noise, we can clearly see the memorization effect of networks. i.e., test accuracy of Standard first reaches a very high level and then gradually decreases due to memorization of noisy labels. Thus, a robust training approach should alleviate or even stop the decreasing trend in test accuracy. On this point, the proposed approach Co-matching shows a clear advantage over other close-related methods, especially in the later stages of learning with noisy labels. The superior performance of Co-matching demonstrates that it prevents memorization of noisy labels throughout the whole training procedure, which consistently verifies our gradient analysis in Section 2.5.3.

We also plot label precision vs. epochs at the bottom row of Figure 2.3 and Figure 2.4. Only Decoupling, Co-teaching, Co-teaching+, JoCoR and Co-matching are considered here, as these methods include sample selection during training. First, we can see Co-matching, JoCoR and Co-teaching can successfully pick clean instances out in Symmetric-20 %, Symmetric-50% and Asymmetric-40% cases. Note that Co-matching not only reaches high label precision in these three cases but also performs better and better with the increase of epochs. Decoupling and Co-teaching+ fail in selecting clean samples, because "Disagreement" strategy does not guarantee to select clean samples, as mentioned in Section 2.4. However, an interesting phenomenon is that high label precision does not necessarily lead to high test accuracy under high-levels of label noise. For example, in Symmetric-80% case, the label precision of Co-matching is much lower than Co-teaching and JoCoR, while the test accuracy is higher than Co-teaching and JoCoR. Similarly, in all noise rates cases on CIFAR-100, Co-teaching has much higher label precision than Co-teaching+, while the test

Methods	*best*	*last*
Standard	67.74	66.95
Decoupling	67.71	66.78
Co-teaching	69.05	68.99
Co-teaching+	67.84	67.68
JoCoR	70.30	69.79
SL	69.22	67.97
GCE	69.59	68.81
APL	69.92	68.84
Co-matching (Ours)	**71.16**	**70.78**

Table 2.2: Test accuracy (%) on Clothing1M with ResNet18. Bold indicates best performance.

accuracy of Co-teaching is lower than Co-teaching+. The subset of clean samples selected by small-loss rule may not rich enough to generalize effectively to held-out data, we believe the samples near the margin with relatively larger loss contribute more towards improving the model's generalization.

Results on real-world dataset Clothing1M. As shown in Table 2.2, *best* denotes the epoch where the validation accuracy is optimal, and *last* denotes the test accuracy at the end of training. Co-matching outperforms the state-of-the-art methods by a large margin on both *best* and *last*, e.g., improving the accuracy from 66.95% to 70.78% over Standard, better than best baseline JoCoR by 0.99%. This verifies the effectiveness of Co-matching against real-world label noise.

2.6.3 Composition of Data Augmentation

We study the impact of data augmentation systematically by considering several common augmentations. One type of augmentation involves spatial/geometric transformation, such as cropping, flipping, rotation and cutout. The other type of augmentation involves appearance transformation, such as color distortion (e.g. brightness

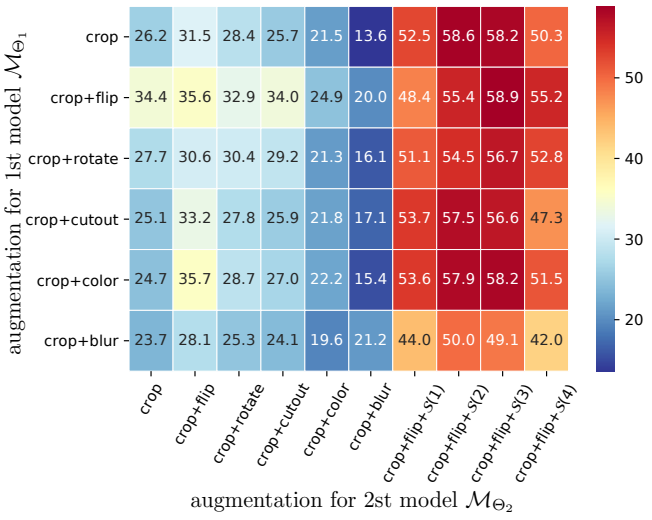


Figure 2.5: Test accuracy(%) over various combinations of augmentations on CIFAR-10 with Symmetric-80% label noise.

and contrast) and Gaussian blur. Since Clothing1M images are of different sizes, we always use cropping as a base transformation. We explore various "weak" augmentation by combining cropping with other augmentations. As for "strong" augmentation, we use RandAugment (Cubuk *et al.*, 2020), which randomly select M transformations from a set $\mathcal{S}$ for each sample in a mini-batch. We denote RandAugment as $\mathcal{S}(M)$. In our experiment, $\mathcal{S} = \{$*Contrast*, *Equalize*, *Invert*, *Rotate*, *Posterize*, *Solarize*, *Color*, *Brightness*, *Sharpness*, *ShearX*, *ShearY*, *Cutout*, *TranslateX*, *TranslateY*, *Gaussian Blur*$\}$.

Figure 2.5 shows the results under composition of transformations. We observe that using "weak" augmentation for both models does not work much better than simple cropping. However, the performance of Co-matching benefits a lot by using stronger augmentations (e.g. add $\mathcal{S}(2)$ and $\mathcal{S}(3)$) on the second network $\mathcal{M}_{\Theta_2}$. We conclude that in the extreme label noise case, our loss function requires using stronger

augmentation on model $\mathcal{M}_{\Theta_2}$ to constantly achieve its ensemble effect.

2.6.4 Ablation Study

In this section, we perform an ablation study to analyze the effect of each component in Co-matching, including the use of two networks, the use of joint update, the use of matching loss and the use of weak and strong augmentations. The experiments are conducted on CIFAR-10 with two cases: Symmetric-50% and Symmetric-80%. To verify the effect of using two networks and the use of joint update, we introduce Standard enhanced by "small-loss" selection (abbreviated as Standard+), Co-teaching and JoCoR to join the comparison. Besides, we simply set $\lambda = 0$ in Eq. (2.3) to see the influence of removing the matching loss (abbreviated as Co-matching-).

The results of their test accuracy vs. epochs are shown in Figure 2.6. In Symmetric-50% case, both Co-teaching and Standard+ keep a downward tendency after increasing to the highest point, which indicates they are still prone to memorizing noisy labels even with "small-loss" update. It also verifies the effect of using two networks as Co-teaching performs better than Standard+. JoCoR consistently outperforms Co-teaching, which verifies the conclusion in (Wei *et al.*, 2020) that joint-update is more efficient than cross-update. However, things start to change in Symmetric-80% case. Co-teaching and Standard+ remain the same trend as these for Symmetric-50% case, but JoCoR performs unstable and even worse than Co-teaching and Standard+. This is likely because that, JoCoR uses the Jensen-Shannon (JS) Divergence to minimize the difference between two networks, resulting in the over consensus of two networks.

In both noise rates cases, Co-matching consistently outperforms Co-matching- and

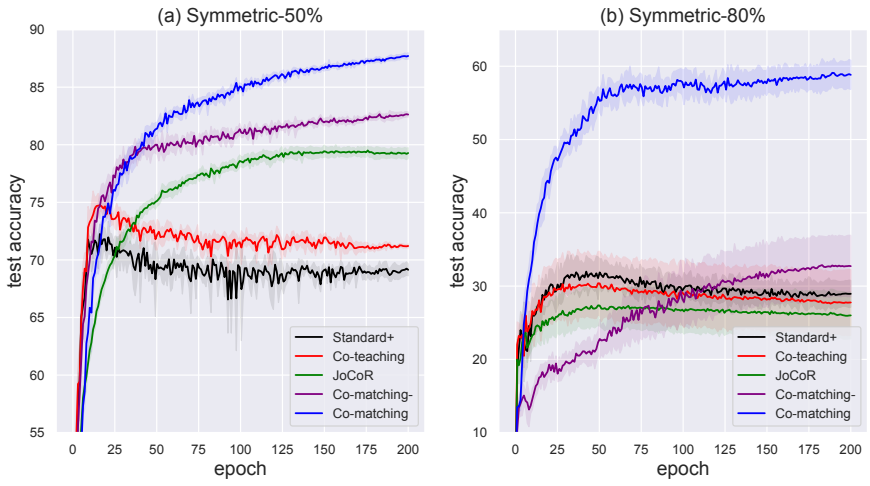


Figure 2.6: Results of ablation study on CIFAR-10

other methods, which validates that the use of matching loss can strongly prevent neural networks from memorizing noisy labels. To show the effect of weak and strong augmentations, we evaluate the state-of-art methods with the same augmentation strategy (i.e. weak and strong for each network respectively) as Co-matching. We conduct the experiments on CIFAR-10 with the hardest Symmetric-80% label noise. Figure 2.7 (a) shows the results. We observe that using weak and strong augmentations may not promise to improve performance for other methods. Co-teaching+ even performs worse. Figure 2.7 (b) shows the influence of λ. A larger λ gets a better accuracy in Symmetric-80% case. More results on hyperparameter sensitivity can be found in Appendix 2.8.2.

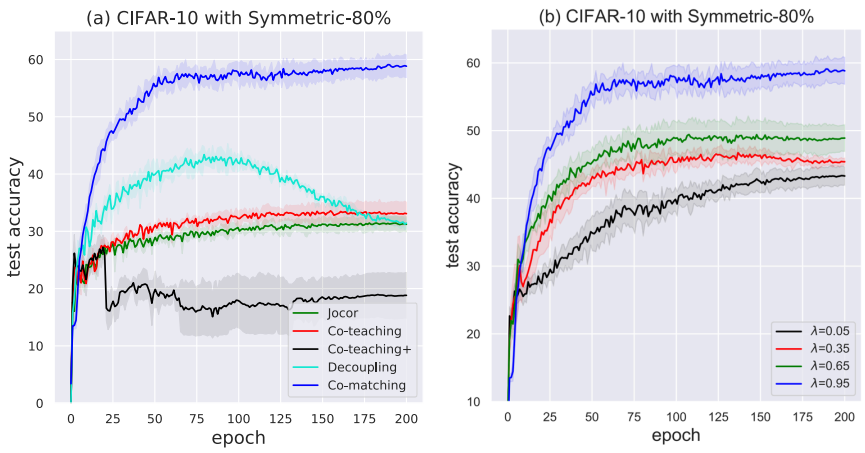


Figure 2.7: (a) Test accuracy of existing methods with same weak and strong augmentations. (b) Test accuracy of Co-matching with different hyperparameter λ.

2.7 Conclusion

In this paper, we identify the deficiencies of existing approaches and introduce a method for deep learning with noisy labels. Our method uses two networks with different strengths of augmented inputs to keep divergence and to achieve better ensemble effect. To avoid the influence of noisy labels, we introduce an unsupervised matching loss for knowledge distillation. In addition, we fit a mixture model to sample loss distribution to select the clean samples without the need of known noise rates. We provide the theoretical analysis on our loss functions and demonstrate the effectiveness of Co-matching on both benchmark and real-world datasets. We believe Co-matching is a promising framework for training robust DNNs against noisy labels from web services.

2.8 Appendix

2.8.1 Details of Datasets and Simulated Noise

For Clothing1M, it contains 1 million images of clothing obtained from online shopping websites with 14 classes: T-shirt, Shirt, Knitwear, Chiffon, Sweater, Hoodie, Windbreaker, Jacket, Down Coat, Suit, Shawl, Dress, Vest, and Underwear. The labels are generated by using the surrounding texts of the images that are provided by the sellers, and thus contain many wrong labels. The overall label noise level of Clothing1M is estimated at 38.46%, with some pairs of classes frequently confused with each other (e.g. Knitwear and Sweater). Note that we only use 14k and 10k clean data for validation and test. The 50k clean training data is not required during the training. As for simulating label noise, Figure 2.8 shows an example of noise transition matrix Q. As for simulated label noise, specifically, for CIFAR-10, the asymmetric noisy labels are generated by flipping *truck* $\rightarrow$ *automobile*, *bird* $\rightarrow$ *airplane*, *deer* $\rightarrow$ *horse* and *cat* $\leftrightarrow$ *dog*. For CIFAR-100, the noise flips each class into the next, circularly within super-classes.

2.8.2 Hyperparameter Sensitivity

Co-matching only has one hyperparameter λ to control the importance weights of classification loss and matching loss. To tune the hyperparameter λ in our loss function Eq. (2.3), we search it in $[0.05, 0.35, 0.65, 0.95]$ with a noisy validation set for optimal performance due to the reliability of noisy validation set (Chen *et al.*, 2021b).

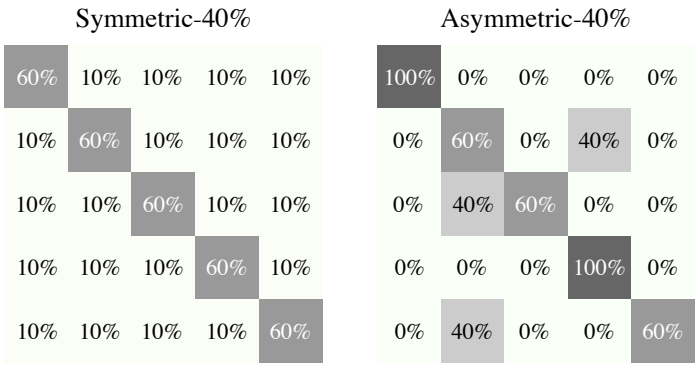


Figure 2.8: Example of noise transition matrix Q (taking 5 classes and noise ratio 0.4 as an example).

Figure 2.9 shows the influence of λ on CIFAR-10. In the CIFAR-10 with Symmetric-50% noise case, $\lambda = 0.35$ returns the best accuracy. Larger or smaller λ hurt the performance. In the CIFAR-10 with Symmetric-80% noise case, a larger λ gets a better accuracy, and $\lambda = 0.95$ achieves the best performance. It verifies the motivation of our loss function: under high-levels of label noise, the model is hard to get enough supervision from noisy labels if we only use the classification loss, more weights on matching loss is required to achieve good performance.

2.8.3 Comparision with Existing Methods

We compare Co-matching to other existing methods in Table 2.8.3.

2.8.4 Hard Pseudo-labeling in Matching Loss

Using hard pseudo-labeling for matching loss helps the model converge. We report the results in Table 2.8.3. We find that when the noise reaches to 80%, the Co-matching

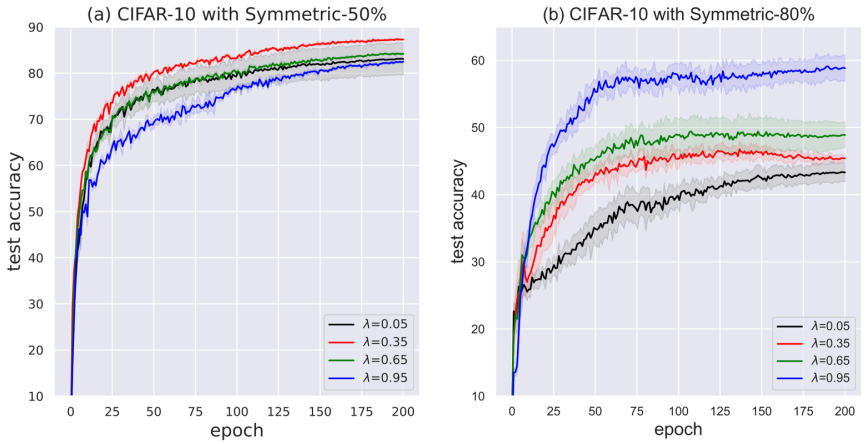


Figure 2.9: Results of Co-matching with different λ on CIFAR-10 with 50% and 80% symmetric label noise.

does not converge with soft pseudo-labeling. In order to improve the stability of our approach, we use hard pseudo-labeling for matching loss in Eq. (2.5).

	Decoupling	Co-teaching	Co-teaching+	JoCoR	Co-matching
cross update	×	√	√	×	×
joint update	×	×	×	√	√
divergence	√	×	√	×	√
augmentation anchoring	×	×	×	×	√
noise rate	×	√	√	√	×

Table 2.3: Comparison of state-of-the-art and related techniques with our approach. In the first column, "cross update": updating parameters in a cross manner instead of a parallel manner; "joint update": updating the two networks parameters jointly. "divergence": keeping two classifiers diverged during the whole training procedure. "augmentation anchoring": encouraging the predictions of a strongly-augmented image to be close to the predictions from a weakly-augmented version of the same image. "noise rate": need a ground truth noise rate or an estimated noise rate.

	Hard pseudo-labeling	Soft pseudo-labeling
Symmetric-20%	90.07 ± 0.16	89.81 ± 0.29
Symmetric-80%	58.86 ± 1.54	9.98 ± 0.00

Table 2.4: Average test accuracy (%) on CIFAR-10.

Chapter 3

SELC: Self-Ensemble Label Correction Improves Learning with Noisy Labels

3.1 Citation and Main Contributor

Lu, Yangdi and He, Wenbo. "SELC: Self-Ensemble Label Correction Improves Learning with Noisy Labels." Published at International Joint Conference on Artificial Intelligence (IJCAI) 2022, Pages 3278-3284. https://doi.org/10.24963/ijcai.2022/455
The main contributor to this paper is the first author - Yangdi Lu (contributes more than 80%).

3.2 Copyright

3.3 Introduction

The recent success of deep neural networks (DNNs) for vision tasks owes much to the availability of large-scale, correctly annotated datasets. However, obtaining such high-quality datasets can be extremely expensive, and sometimes even impossible. The common approaches, such as web queries (Li *et al.*, 2017) and crowdsourcing (Song *et al.*, 2019), can easily provide extensive labeled data, but unavoidably introduce *noisy labels*. Existing studies (Arpit *et al.*, 2017; Zhang *et al.*, 2018a) have demonstrated that DNNs can easily overfit noisy labels, which deteriorates the generalization performance. Thus, it is essential to develop noise-robust algorithms for learning with noisy labels.

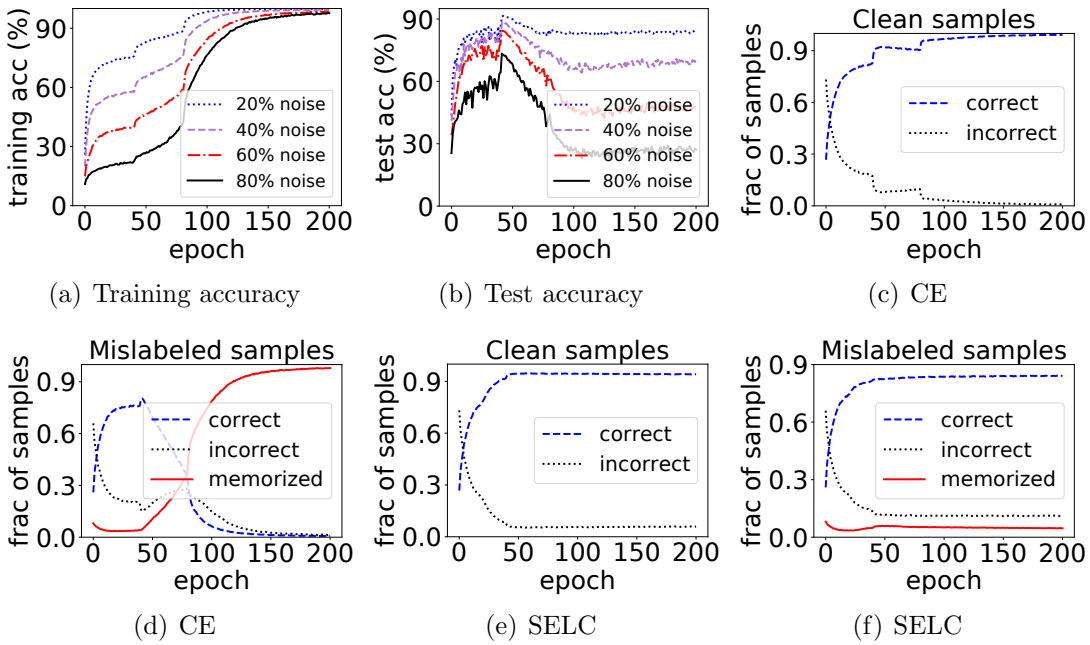
Given a noisy training set consisting of clean samples and mislabeled samples, a common category of approaches (Reed *et al.*, 2015; Arazo *et al.*, 2019; Zhang *et al.*, 2020) to mitigating the negative influence of noisy labels is to identify and correct the mislabeled samples. However, the correction procedure in these methods only updates the noisy labels using the model prediction from the most recent training epoch directly, thus it may suffer from the false correction as the model predictions for noisy samples tend to fluctuate. Take a bird image mislabeled as an airplane as an example. During the training, the clean bird samples would encourage the model to predict a given bird image as a bird, while the bird images with airplane labels regularly pull the model back to predict the bird as an airplane. Hence, the model prediction gathered in one training epoch may change back and forth between bird and airplane, resulting in false correction.

We investigate the reason for performance degradation by analyzing the memorization behavior of the DNNs models. We observe that there exists a *turning point*

during training. Before the turning point, the model only learns from easy (clean) samples, and thus model prediction is likely to be consistent with clean samples. After the turning point, the model increasingly memorizes hard (mislabeled) samples. Hence model prediction oscillates strongly on clean samples. Triggered by this observation, we seek to make the model retain the early-learning memory for consistent predictions on clean samples even after the turning point.

In this paper, we propose self-ensemble label correction (SELC), which potentially corrects noisy labels during training thus preventing the model from being affected by the noisy labels. SELC leverages the knowledge provided in the model predictions over historical training epochs to form a consensus of prediction (ensemble prediction) before the turning point. We demonstrate that combining ensemble prediction with the original noisy label leads to a better target. Accordingly, the model is gradually refined as the targets become less noisy, resulting in improving performance. However, it is challenging to find the turning point. Existing works estimate the turning point based on a test set or noise information, which are unobservable in practice. We propose a metric to estimate the turning point only using training data, allowing us to select a suitable initial epoch to perform SELC. Overall, our contributions are summarized as follows:

- We propose a simple and effective label correction method SELC based on self-ensembling.
- We design an effective metric based on unsupervised loss modeling to detect the turning point without requiring the test set and noise information.
- SELC achieves superior results and can be integrated with other techniques such as mixup (Zhang *et al.*, 2018b) to further enhance the performance.



(a) Training accuracy (b) Test accuracy (c) CE

(d) CE (e) SELC (f) SELC

Figure 3.1: Plots (a) and (b) show the training and test accuracy on CIFAR-10 with different ratios of label noise using cross-entropy (CE) loss. We investigate the memorization behavior of DNNs on CIFAR-10 with 60% label noise using CE loss and SELC. Plot (c) and (e) show the fraction of clean samples that are predicted correctly (blue) and incorrectly (black). Plot (d) and (f) show the fraction of mislabeled samples that are predicted correctly (blue), *memorized* (i.e. the prediction equals to the wrong label, shown in red), and incorrectly predicted as neither the true nor the given wrong label (black). Compared to CE, SELC effectively prevents memorization of mislabeled samples and refines the model to attain correct predictions on both clean and mislabeled samples.

3.4 Related Work

We briefly discuss the existing methods that do not require a small set of clean data (as opposed to (Xiao *et al.*, 2015)).

Robust loss functions. Some methods aim to develop loss functions that are robust to label noise, including GCE (Zhang and Sabuncu, 2018), $\mathcal{L}_{\text{DMI}}$ (Xu *et al.*, 2019), SCE (Wang *et al.*, 2019) and NCE (Ma *et al.*, 2020). **Loss correction.**

Patrini *et al.* (2017) focus on correcting the loss function explicitly by estimating the noise transition matrix. **Sample selection.** Co-training-based methods (Han *et al.*, 2018; Lu *et al.*, 2022) maintain two networks, and each network is trained on low-risk samples which are selected by its peer network based on the small-loss criterion. **Regularization.** These methods (Liu *et al.*, 2020; Lu *et al.*, 2021) prevent memorization of mislabeled samples by using a regularizer. **Label filtering.** SELF (Nguyen *et al.*, 2020) filters the mislabeled samples by ensemble predictions to improve the performance. **Label correction.** Joint Opt (Tanaka *et al.*, 2018) and PENCIL (Yi and Wu, 2019) replace the noisy labels with soft (i.e. model probability) or hard (i.e. to one-hot vector) pseudo-labels. Bootstrap (Reed *et al.*, 2015) and M-correction (Arazo *et al.*, 2019) correct the labels by using a convex combination of noisy labels and the model prediction. PLC (Zhang *et al.*, 2020) updates the noisy labels of high confident samples with model predictions.

Our method is related to label correction. Compared with existing methods, we focus on using ensemble prediction based on historical model outputs to correct the noisy labels, rather than only using prediction from the most recent training epoch. Our approach is straightforward and yields superior performance. Furthermore, our technique can be employed as an add-on component to further enhance the other approaches in challenging cases.

3.5 Preliminaries

Supervised Classification. Considering a supervised classification problem with C classes, suppose $\mathcal{X} \in \mathbb{R}^d$ be the input space, $\mathcal{Y} \in \{0,1\}^C$ is the ground-truth label space in a one-hot manner. In practice, the joint distribution $\mathcal{P}$ over $\mathcal{X} \times \mathcal{Y}$ is

unknown. We have a training set $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ which are independently sampled from $\mathcal{P}$. Assume a mapping function class $\mathcal{F}$ wherein each $f : \mathcal{X} \to \mathbb{R}^C$ maps the input space to C-dimensional score space, we seek $f^* \in \mathcal{F}$ that minimizes an empirical risk $\frac{1}{N}\sum_{i=1}^N \ell(\boldsymbol{y}_i, f(\boldsymbol{x}_i))$ for a certain loss function ℓ.

Learning with Noisy Labels. Our goal is to learn from a noisy training distribution $\mathcal{P}_\eta$ where the labels are corrupted, with probability η, from their true distribution $\mathcal{P}$. Given a noisy training set $\hat{D} = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^N$, the observable noisy label $\hat{\boldsymbol{y}}_i$ has a probability of η to be incorrect. Suppose the mapping function f is a deep neural network classifier parameterized by Θ. f maps an input $\boldsymbol{x}_i$ to C-dimensional logits $\boldsymbol{z}_i = f(\boldsymbol{x}_i, \Theta)$. We obtain conditional probability of each class by using a softmax function $\mathcal{S}(\cdot)$, thus $\boldsymbol{p}_i = \mathcal{S}(\boldsymbol{z}_i)$. Then the empirical risk on $\hat{D}$ using cross-entropy loss is

$$\mathcal{L}_{\text{ce}} = \frac{1}{N}\sum_{i=1}^{N} \ell_{\text{ce}}(\hat{\boldsymbol{y}}_i, \boldsymbol{p_i}) = -\frac{1}{N}\sum_{i=1}^{N} (\hat{\boldsymbol{y}}_i)^\top \log(\boldsymbol{p}_i). \tag{3.1}$$

When optimizing $\mathcal{L}_{\text{ce}}$ by stochastic gradient descent (SGD), the DNNs have been observed to completely fit the training set including mislabeled samples eventually (see Figure 3.1 (a)), resulting in the test performance degradation in the later stage of training (see Figure 3.1 (b)).

Noise Models. The generation of real-world label noise is unpredictable, a common methodology to cope with noisy labels is to posit a noise model and design robust algorithms under this model. Then we evaluate the algorithms on the real-world datasets to see their effectiveness. A common noise model is class-conditional noise

(Natarajan *et al.*, 2013), wherein label noise is independent of input features and true label is corrupted by either a *symmetric* or *asymmetric* noise transition matrix (details are in Section 3.7.1). Recently, another label noise model, named instance-dependent noise (Zhang *et al.*, 2020; Chen *et al.*, 2021a), is proposed, in which the noise not only depends on the class but also the input feature.

3.6 Our Method

3.6.1 Memorization Behavior

Our motivation stems from the memorization behavior of DNNs when trained with noisy labels. In Figure 3.1 (c), we observe that for clean samples, the model predicts them correctly with the increase of epochs. For mislabeled samples in Figure 3.1 (d), the model predicts the true labels correctly for most mislabeled samples in the early stage (high blue line), even though the model begins making incorrect predictions because of the memorization of wrong labels (increasing red line). Since the model predictions are relatively correct for both mislabeled and clean samples in the early stage, can these reliable model predictions help correct the noisy labels?

3.6.2 Ensemble Prediction

To alleviate the impact of noisy labels, existing work Bootstrap (Reed *et al.*, 2015) proposes to generate soft target by interpolating between the original noisy distributions and model predictions by $\beta\hat{\boldsymbol{y}} + (1-\beta)\boldsymbol{p}$, where β weights the degree of

interpolation. Thus the cross-entropy loss using Bootstrap becomes

$$\mathcal{L}_{\text{bs}} = -\frac{1}{N}\sum_{i=1}^{N}\Big(\beta\hat{\boldsymbol{y}}_i + (1-\beta)\boldsymbol{p}_i\Big)^{\top}\log(\boldsymbol{p}_i). \tag{3.2}$$

However, applying a static weight (e.g. $\beta = 0.8$) to the prediction limits the correction of a hypothetical noisy label. Although another work M-correction (Arazo *et al.*, 2019) makes β dynamic for different samples, the one-step correction based solely on the model predictions at the most recent training epoch still easily incurs false correction.

Since the predictions gathered in a single training epoch for correction is sub-optimal, we generate the ensemble prediction $\tilde{\boldsymbol{p}}$ for each sample, aggregating the predictions over multiple previous epochs by exponential moving average. Let's denote the model prediction in epoch k as $\boldsymbol{p}_{[k-1]}$. In epoch k, we have ensemble prediction

$$\tilde{\boldsymbol{p}}_{[k]} = \begin{cases} \mathbf{0} & \text{if } k = 0 \\ \alpha\tilde{\boldsymbol{p}}_{[k-1]} + (1-\alpha)\boldsymbol{p}_{[k]}, & \text{if } k > 0 \end{cases} \tag{3.3}$$

where $0 \leq \alpha < 1$ is the momentum. Based on the Eq. (3.3), we can derive the ensemble prediction in k-th epoch as $\tilde{\boldsymbol{p}}_{[k]} = \sum_{j=1}^{k}(1-\alpha)\alpha^{k-j}\boldsymbol{p}_{[j]}$. Although ensemble prediction requires a new hyperparameter α and auxiliary memory to record, it maintains a more stable and accurate prediction, especially for mislabeled samples.

3.6.3 Self-Ensemble Label Correction

We seek to utilize the ensemble predictions to progressively enhance the targets in loss function. There are two options to be considered.

- *Option I.* Directly use ensemble prediction as the target.

- *Option II.* Preserve the original noisy label, and combine it with ensemble prediction as the target.

The first option is widely adopted in semi-supervised learning, as the ensemble prediction learned from the labeled inputs can be used as targets for the unlabeled inputs. However, in the noisy labels setting, the model needs supervisions from noisy labels as no extra clean samples are provided. We would compare these two options in Section 3.7.4. In SELC, we choose the second option. Specifically, for each training sample, we initialize the soft target $\boldsymbol{t}$ using original noisy label $\hat{\boldsymbol{y}}$. Then we update $\boldsymbol{t}$ in each training epoch k by

$$\boldsymbol{t}_{[k]} = \begin{cases} \hat{\boldsymbol{y}} & \text{if } k = 0 \\ \alpha \boldsymbol{t}_{[k-1]} + (1-\alpha)\boldsymbol{p}_{[k]}. & \text{if } k > 0 \end{cases} \tag{3.4}$$

Based on the Eq. (3.4), we rewrite above equation as

$$\boldsymbol{t}_{[k]} = \underbrace{\alpha^k \hat{\boldsymbol{y}}}_{\text{first term}} + \underbrace{\sum_{j=1}^{k} (1-\alpha)\alpha^{k-j}\boldsymbol{p}_{[j]}}_{\text{second term}}. \tag{3.5}$$

The first term preserves the original noisy labels with exponential decaying weights α^k. The second term is exactly the ensemble prediction at epoch k. Therefore, in training epoch k, the loss of SELC becomes

$$\begin{aligned} \mathcal{L}_{\text{selc}} &= -\frac{1}{N}\sum_{i=1}^{N}\Big(\alpha^k \hat{\boldsymbol{y}}_{\boldsymbol{i}} + \sum_{j=1}^{k}(1-\alpha)\alpha^{k-j}\boldsymbol{p}_{\boldsymbol{i}[\boldsymbol{j}]}\Big)^{\top}\log(\boldsymbol{p}_i) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\alpha^k(\hat{\boldsymbol{y}}_{\boldsymbol{i}})^{\top}\log(\boldsymbol{p}_i) - \frac{1}{N}\sum_{i=1}^{N}\Big(\sum_{j=1}^{k}(1-\alpha)\alpha^{k-j}\boldsymbol{p}_{\boldsymbol{i}[\boldsymbol{j}]}\Big)^{\top}\log(\boldsymbol{p}_i) \end{aligned} \tag{3.6}$$

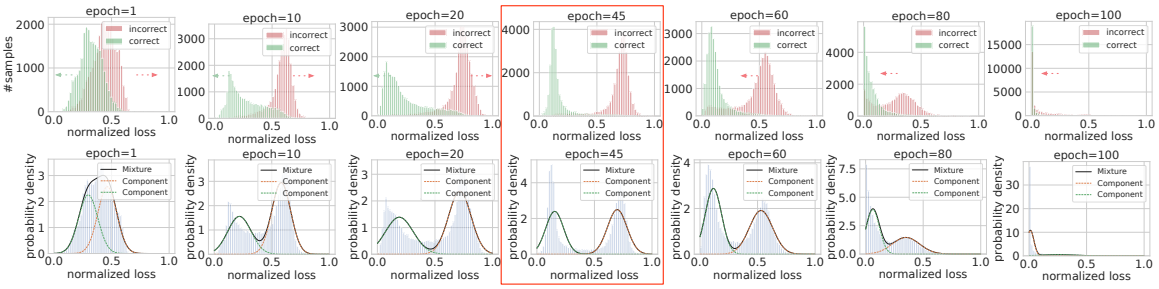


Figure 3.2: We train ResNet34 on the CIFAR-10 with 60% symmetric noise using CE loss and investigate the loss distribution. Top row: The normalized loss distribution over different training epochs. Bottom row: The corresponding mixture model after fitting a two-component GMM to loss distribution. Two components gradually separate at the beginning and start to merge after the turning point (red box).

where $\boldsymbol{p_{i[j]}}$ denotes the model prediction in epoch j for input $\boldsymbol{x}_i$. The first loss term is actually the cross-entropy loss $\mathcal{L}_{\text{ce}}$ but weighed by α^k. With the increase of training epoch k, α^k becomes smaller. Thus $\mathcal{L}_{\text{selc}}$ is less and less reliant on original noisy labels. The second loss term maintains an exponential moving average of historical prediction as target, and penalizes model predictions that are inconsistent with this target. As a consequence, SELC effectively prevents memorization of mislabeled samples (low red line in Figure 3.1 (f)) and attains superior performance.

3.6.4 Estimation of Turning Point

In semi-supervised learning, the initial epoch to perform ensemble prediction is not crucial as the supervision from the clean set guides the model to predict consistent prediction throughout the training. Comparatively, in our scenario, the model would overfit to noisy labels, causing the model predictions to deteriorate. Therefore, it is essential to select the initial epoch in SELC before the turning point T, at which the

model starts to memorize mislabeled samples.

We can clearly observe the occurrence of turning point by monitoring the test accuracy drop (Figure 3.1 (b)). However, the test set is unobservable in practice. The way to accurately identify the turning point without a test set and noise information remains challenging and underexplored.

In this paper, we propose three metrics and choose the optimal one to estimate the turning point by modeling training samples' loss distribution without requiring a clean test set. Due to the memorization behavior of DNNs, the clean samples tend to have smaller loss values than the mislabeled samples in early stage. We analyze the normalized loss distribution over different training epochs in Figure 3.2 top row. Intriguingly, the two distributions are merged at the initialization, then start to separate, but resume merging after the turning point. Therefore, we propose to estimate the turning point by finding the epoch that has the largest distance between two distributions. To model these two distributions, we use two unsupervised learning approaches: Gaussian Mixture Model (GMM) (Permuter *et al.*, 2006) and K-Means.

Metric 1 and Metric 2. We fit a two-component GMM to loss distribution (in Figure 3.2 bottom row). The probability density function (pdf) of GMM with M components on the per sample loss value ℓ can be defined as $P(\ell) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\ell \mid \mu_m, \sigma_m^2)$, $\sum_{m=1}^{M} \pi_m = 1$, where π_m is the coefficient for the linear convex combination of each individual pdf $\mathcal{N}(\ell \mid \mu_m, \sigma_m^2)$. We use the Expectation-Maximization (EM) algorithm to estimate the π_m, μ_m and σ_m^2. For Metric 1, we directly calculate the distance between two components by

$$\mathcal{M}_1 = \mid \mu_1 - \mu_2 \mid . \tag{3.7}$$

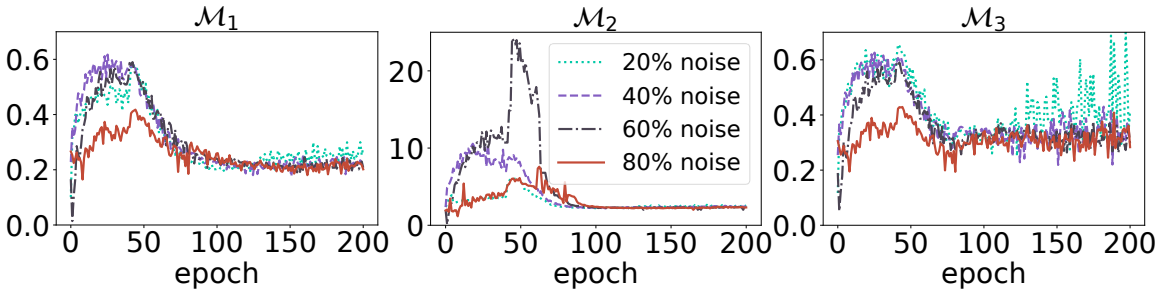


Figure 3.3: Three metrics on CIFAR-10 with different ratios of noise.

For Metric 2, we calculate the Kullback–Leibler (KL) divergence of two components as distance.

$$\mathcal{M}_2 = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \tag{3.8}$$

Metric 3. We fit two clusters by K-Means on the loss distribution. Then we calculate the distance between two cluster centroids $\mathcal{S}_1$ and $\mathcal{S}_2$ as the Metric 3.

$$\mathcal{M}_3 = \mid \mathcal{S}_1 - \mathcal{S}_2 \mid . \tag{3.9}$$

When we train the DNNs with noisy labels, we monitor these three metrics. Once they achieve the maximum value, the corresponding epoch is likely to be the turning point T. We compare three metrics on CIFAR-10 with label noise in Figure 3.3. $\mathcal{M}_1$ is the most reliable and stable one since its corresponding epoch of maximum value precisely aligns with the epoch when test accuracy starts to drop in Figure 3.1 (b) in all noise cases. We put pseudocode of SELC in Algorithm 2.

Algorithm 2: SELC pseudocode.

Input: DNNs $f(\Theta)$, training data $\hat{D} = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^{N}$, Estimated turning point T, total epoch $T_{\max}$, hyperparameter α
Let target $\boldsymbol{t} = \hat{\boldsymbol{y}}$;
Select an initial epoch $T_e < T$ (e.g. $T_e = T - 10$) ;
while *epoch* $e < T_{max}$ **do**
    if *epoch* $e < T_e$ **then**
        Train $f(\Theta)$ by CE loss in Eq. (3.1) using SGD
    else
        Update $\boldsymbol{t}$ by Eq. (3.5)
        Train $f(\Theta)$ by SELC loss in Eq. (3.6) using SGD
Output: Optimized DNN $f(\Theta^*)$

3.7 Experiments

This section, first, investigates the effectiveness of the proposed SELC for classification with class-conditional noise (Section 3.7.1), instance-dependent noise (Section 3.7.2) and real-world noise (Section 3.7.3). This is followed by several empirical analyses (Section 3.7.4) to shed light on SELC.

3.7.1 Class-conditional Label Noise

Datasets and Networks. We conduct the experiments with class-conditional label noise on CIFAR-10 and CIFAR-100 (Krizhevsky *et al.*, 2009). Given these two datasets are initially clean, we follow (Patrini *et al.*, 2017) to inject noise by label transition matrix $\mathbf{Q}$, where $\mathbf{Q}_{ij} = \Pr[\hat{y} = j \mid y = i]$ denotes the probability that noisy label $\hat{y}$ is flipped from true label y. We evaluate SELC in two types of noise: *symmetric* and *asymmetric*. Symmetric noise is generated by replacing the labels for a percentage of the training data with all possible labels uniformly. Asymmetric noise

Dataset	CIFAR-10					CIFAR-100				
Class-conditional noise type	symm				asymm	symm				asymm
Method/Noise ratio	20%	40%	60%	80%	40%	20%	40%	60%	80%	40%
Cross Entropy	86.98 ± 0.12	81.88 ± 0.29	74.14 ± 0.56	53.82 ± 1.04	80.11 ± 1.44	58.72 ± 0.26	48.20 ± 0.65	37.41 ± 0.94	18.10 ± 0.82	42.74 ± 0.61
Bootstrap (Reed *et al.*, 2015)	86.23 ± 0.23	82.23 ± 0.37	75.12 ± 0.56	54.12 ± 1.32	81.21 ± 1.47	58.27 ± 0.21	47.66 ± 0.55	34.68 ± 1.10	21.64 ± 0.97	45.12 ± 0.57
Forward (Patrini *et al.*, 2017)	87.99 ± 0.36	83.25 ± 0.38	74.96 ± 0.65	54.64 ± 0.44	83.55 ± 0.58	39.19 ± 2.61	31.05 ± 1.44	19.12 ± 1.95	8.99 ± 0.58	34.44 ± 1.93
GCE (Zhang and Sabuncu, 2018)	89.83 ± 0.20	87.13 ± 0.22	82.54 ± 0.23	64.07 ± 1.38	76.74 ± 0.61	66.81 ± 0.42	61.77 ± 0.24	53.16 ± 0.78	29.16 ± 0.74	47.22 ± 1.15
Mixup (Zhang *et al.*, 2018b)	93.58	89.46	78.32	66.32	81.66	69.31	58.12	41.10	18.77	49.61
Joint Opt (Tanaka *et al.*, 2018)	92.25	90.79	86.87	69.16	-	58.15	54.81	47.94	17.18	-
PENCIL (Yi and Wu, 2019)	-	-	-	-	91.01	-	69.12 ± 0.62	57.70 ± 3.86	fail	63.61 ± 0.23
NLNL (Kim *et al.*, 2019)	94.23	92.43	88.32	-	89.86	71.52	66.39	56.51	-	45.70
SCE (Wang *et al.*, 2019)	89.83 ± 0.20	87.13 ± 0.26	82.81 ± 0.61	68.12 ± 0.81	82.51 ± 0.45	70.38 ± 0.13	62.27 ± 0.22	54.82 ± 0.57	25.91 ± 0.44	69.32 ± 0.87
M-correction (Arazo *et al.*, 2019)	-	92.30	86.10	74.10	-	-	70.10	59.50	**39.50**	-
DAC (Thulasidasan *et al.*, 2019a)	92.91	90.71	86.30	74.84	-	73.55	66.92	57.17	32.16	-
SELF (Nguyen *et al.*, 2020)	-	91.13	-	63.59	-	-	66.71	-	35.56	-
NCE+RCE (Ma *et al.*, 2020)	-	86.02 ± 0.09	79.78 ± 0.50	52.71 ± 1.90	79.59 ± 0.40	-	59.48 ± 0.56	47.12 ± 0.62	25.80 ± 1.12	46.69 ± 0.96
ELR (Liu *et al.*, 2020)	91.16 ± 0.08	89.15 ± 0.17	86.12 ± 0.49	73.86 ± 0.61	90.12 ± 0.47	74.21 ± 0.22	68.28 ± 0.31	59.28 ± 0.67	29.78 ± 0.56	73.26 ± 0.64
SELC (Ours)	93.09 ± 0.02	91.18 ± 0.06	87.25 ± 0.09	74.13 ± 0.14	91.05 ± 0.11	73.63 ± 0.07	68.46 ± 0.10	59.41 ± 0.06	32.63 ± 0.06	70.82 ± 0.09
SELC+ (Ours)	**94.97 ± 0.04**	**93.12 ± 0.08**	**90.46 ± 0.12**	**78.62 ± 0.24**	**92.92 ± 0.10**	**76.39 ± 0.15**	**71.73 ± 0.11**	**64.49 ± 0.15**	37.18 ± 0.50	**73.58 ± 0.11**

Table 3.1: Test accuracy (%) on CIFAR-10/100 with various ratios of class-conditional label noise injected to the training set. All methods use the same backbone ResNet34. The average accuracy and standard deviation over 3 trials are reported. The best results are in **bold**.

Method/Noise ratio	10%	20%	30%	40%
Cross Entropy	91.25 ± 0.27	86.34 ± 0.11	80.87 ± 0.05	75.68 ± 0.29
Forward (Patrini *et al.*, 2017)	91.06 ± 0.02	86.35 ± 0.11	78.87 ± 2.66	71.12 ± 0.47
Co-teaching (Han *et al.*, 2018)	91.22 ± 0.25	87.28 ± 0.20	84.33 ± 0.17	78.72 ± 0.47
GCE (Zhang and Sabuncu, 2018)	90.97 ± 0.21	86.44 ± 0.23	81.54 ± 0.15	76.71 ± 0.39
DAC (Thulasidasan *et al.*, 2019a)	90.94 ± 0.09	86.16 ± 0.13	80.88 ± 0.46	74.80 ± 0.32
DMI (Xu *et al.*, 2019)	91.26 ± 0.06	86.57 ± 0.16	81.98 ± 0.57	77.81 ± 0.85
SEAL (Chen *et al.*, 2021a)	91.32 ± 0.14	87.79 ± 0.09	85.30 ± 0.01	82.98 ± 0.05
SELC (Ours)	**91.63 ± 0.15**	**88.33 ± 0.16**	**86.28 ± 0.22**	**84.23 ± 0.37**

Table 3.2: Test accuracy (%) on CIFAR-10 under instance-dependent label noise from SEAL with different noise ratios. All the compared methods do not use mixup to boost the performance.

Dataset	Noise	Cross Entropy	Co-teaching+ (Yu *et al.*, 2019)	GCE (Zhang and Sabuncu, 2018)	SCE (Wang *et al.*, 2019)	LRT (Zheng *et al.*, 2020)	PLC (Zhang *et al.*, 2020)	SELC (ours)
CIFAR-10	Type-I (35%)	78.11 ± 0.74	79.97 ± 0.15	80.65 ± 0.39	79.76 ± 0.72	80.98 ± 0.80	82.80 ± 0.27	**86.97 ± 0.15**
	Type-I (70%)	41.98 ± 1.96	40.69 ± 1.99	36.52 ± 1.62	36.29 ± 0.66	41.52 ± 4.53	42.74 ± 2.14	**43.78 ± 2.64**
	Type-II (35%)	76.65 ± 0.57	77.34 ± 0.44	77.60 ± 0.88	77.92 ± 0.89	80.74 ± 0.25	81.54 ± 0.47	**87.06 ± 0.20**
	Type-II (70%)	45.57 ± 1.12	45.44 ± 0.64	40.30 ± 1.46	41.11 ± 1.92	44.67 ± 3.89	46.04 ± 2.20	**46.79 ± 3.06**
	Type-III (35%)	76.89 ± 0.79	78.38 ± 0.67	79.18 ± 0.61	78.81 ± 0.29	81.08 ± 0.35	81.50 ± 0.50	**87.31 ± 0.18**
	Type-III (70%)	43.32 ± 1.00	41.90 ± 0.86	37.10 ± 0.59	38.49 ± 1.46	44.47 ± 1.23	45.05 ± 1.13	**45.57 ± 1.71**
CIFAR-100	Type-I (35%)	57.68 ± 0.29	56.70 ± 0.71	58.37 ± 0.18	55.20 ± 0.33	56.74 ± 0.34	60.01 ± 0.43	**65.72 ± 0.17**
	Type-I (70%)	39.32 ± 0.43	39.53 ± 0.28	40.01 ± 0.71	40.02 ± 0.85	45.29 ± 0.43	45.92 ± 0.61	**49.72 ± 0.15**
	Type-II (35%)	57.83 ± 0.25	56.57 ± 0.52	58.11 ± 1.05	56.10 ± 0.73	57.25 ± 0.68	63.68 ± 0.29	**66.79 ± 0.18**
	Type-II (70%)	39.30 ± 0.32	36.84 ± 0.39	37.75 ± 0.46	38.45 ± 0.45	43.71 ± 0.51	45.03 ± 0.50	**52.65 ± 0.26**
	Type-III (35%)	56.07 ± 0.79	55.77 ± 0.98	57.51 ± 1.16	56.04 ± 0.74	56.57 ± 0.30	63.68 ± 0.29	**66.41 ± 0.17**
	Type-III (70%)	40.01 ± 0.18	35.37 ± 2.65	40.53 ± 0.60	39.94 ± 0.84	44.41 ± 0.19	44.45 ± 0.62	**49.85 ± 0.36**

Table 3.3: Test accuracy (%) on CIFAR under different types of PMD noise with various levels. The average accuracy and standard deviation over 3 trials are reported. All above methods do not use mixup to boost the performance for fair comparison. The best results are in **bold**.

Method	Accuracy
Cross Entropy	79.40 ± 0.14
Nested (Chen *et al.*, 2021c)	81.30 ± 0.60
SELFIE (Song *et al.*, 2019)	81.80 ± 0.09
PLC (Zhang *et al.*, 2020)	83.40 ± 0.43
SELC (ours)	**83.73 ± 0.06**

Table 3.4: The accuracy (%) results on ANIMAL-10N.

Method	Accuracy
Cross Entropy	68.94
Forward (Patrini *et al.*, 2017)	69.84
SEAL (Chen *et al.*, 2021a)	70.63
SCE (Wang *et al.*, 2019)	71.02
LRT (Zheng *et al.*, 2020)	71.74
DMI (Xu *et al.*, 2019)	72.27
ELR (Liu *et al.*, 2020)	72.87
Nested (Chen *et al.*, 2021c)	73.10
PENCIL (Yi and Wu, 2019)	73.49
PLC (Zhang *et al.*, 2020)	**74.02**
SELC (ours)	74.01

Table 3.5: The accuracy (%) results on Clothing1M.

is designed to mimic the structure of real-world label noise, where the annotators are more likely to make mistakes only within very similar classes (e.g. *deer* $\rightarrow$ *horse* and *cat* $\leftrightarrow$ *dog*). We use the ResNet34 He *et al.* (2016a) as backbone for both datasets, and train the model using SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 128. The network is trained for 200 epochs. We set the initial learning rate as 0.02, and reduce it by a factor of 10 after 40 and 80 epochs. We fix

Method	Webvision		ILSVRC12	
	top1	top5	top1	top5
Forward (Patrini *et al.*, 2017)	61.12	82.68	57.36	82.36
Co-teaching (Han *et al.*, 2018)	63.58	85.20	61.48	84.70
Iterative-CV (Chen *et al.*, 2019)	65.24	85.34	61.60	84.98
RSL (Gui *et al.*, 2021)	65.64	85.72	62.04	84.84
CRUST (Mirzasoleiman *et al.*, 2020)	72.40	89.56	67.36	87.84
SELC (ours)	**74.38**	**90.66**	**70.85**	**90.74**

Table 3.6: The accuracy (%) results on (mini) Webvision.

hyperparameter $\alpha = 0.9$. More discussions on α are in Section 3.7.4. Note that we do not perform early stopping since we don't assume the presence of clean validation data. All test accuracy are recorded from the last epoch of training.

Baselines: We compare SELC with the state-of-the-art methods, including Forward (Patrini *et al.*, 2017), GCE (Zhang and Sabuncu, 2018), Mixup (Zhang *et al.*, 2018b), Joint Opt (Tanaka *et al.*, 2018), PENCIL (Yi and Wu, 2019), NLNL (Kim *et al.*, 2019), SCE (Wang *et al.*, 2019), M-correction (Arazo *et al.*, 2019), DAC (Thulasidasan *et al.*, 2019a), SELF (Nguyen *et al.*, 2020), NCE+RCE (Ma *et al.*, 2020) and ELR (Liu *et al.*, 2020).

Improving Other Methods. In comparison to the original noisy labels, we obtain cleaner targets $\boldsymbol{t}$ after using SELC. Therefore, SELC can be easily integrated with other methods. For fair comparison with the existing approaches (e.g. M-correction and NLNL) that use mixup or multiple stages of training to boost the performance, we propose SELC+ which uses the corrected labels from SELC to retrain an initialized DNNs using mixup.

Results. Table 3.1 shows the results on CIFAR with different types and levels of class-conditional label noise. SELC achieves excellent performance compared to the methods that only modify the training loss without extra techniques to boost the performance. When integrated with mixup data augmentation, SELC+ achieves the best performance across most noise ratios, demonstrating the effectiveness of the proposed method on class-conditional label noise.

3.7.2 Instance-dependent Label Noise

Datasets and Networks. We follow the recent works SEAL (Chen *et al.*, 2021a) and PLC (Zhang *et al.*, 2020) to inject instance-dependent label noise to CIFAR. SEAL generates the controllable label noise based on the assumption that 'hard' (low confidence) samples are more likely to be mislabeled. PLC introduces Polynomial Margin Diminishing (PMD) noise which allows arbitrary noise strength in a wide buffer near the decision boundary. For fair comparison with SEAL, we use the same network architecture Wide ResNet28×10. As for PMD noise, we use the same network architecture PreAct ResNet34 as PLC.

Results. Table 3.2 shows the results on instance-dependent label noise from SEAL. Our approach consistently achieves the best generalization performance over different noise ratios. The larger the noise ratio is, the more improvement SELC obtains. Table 3.3 lists the performance of different methods under three types of PMD noise at noise level 35% and 70%. We observe that the proposed method outperforms baselines across different noise settings. When the noise level is high, performances of a few baselines deteriorate and become worse than the standard (CE) approach. In contrast, the improvement of SELC is substantial (∼10% in accuracy) for the more challenging CIFAR-100 with 70% label noise.

3.7.3 Real-world Label Noise

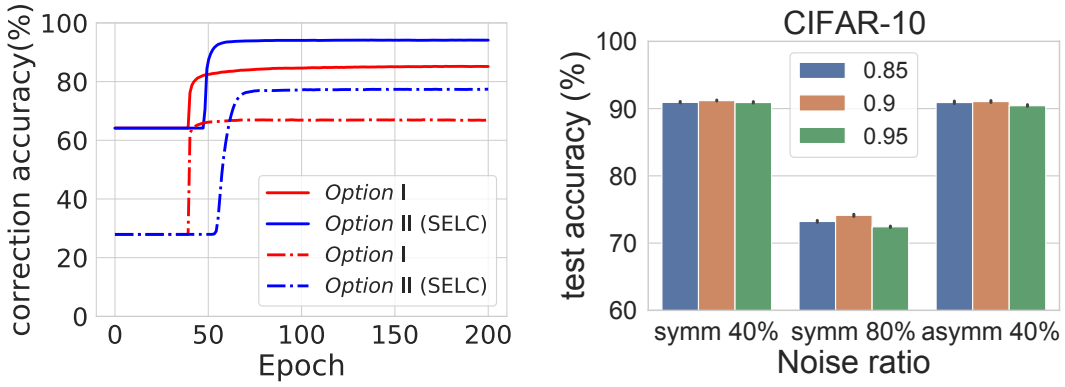
Datasets and Networks. We use ANIMAL-10N (Song *et al.*, 2019), Clothing1M (Xiao *et al.*, 2015) and Webvision (Li *et al.*, 2017) to evaluate the performance of SELC

under the real-world label noise settings. ANIMAL-10N contains human-labeled online images for 10 animals with confusing appearance. The estimated label noise rate is 8%. Clothing1M consists of 1 million images collected from online shopping websites with labels generated from surrounding texts. The estimated label noise rate is 38.5%. WebVision contains 2.4 million images crawled from the web using the 1,000 concepts in ImageNet ILSVRC12. The estimated label noise rate is 20%. For ANIMAL-10N, we use VGG-19 with batch normalization. For Clothing1M, we use ResNet50 pretrained on ImageNet. For Webvision, we use InceptionResNetV2. Note that all the compared method do not use mixup to boost the performance for fair comparison.

Results. Table 3.4, Table 3.5 and Table 3.6 show the results on ANIMAL-10N, Clothing1M and Webvision respectively. On ANIMAL-10N and Webvision, our approach outperforms the existing baselines. On Clothing1M, SELC achieves the comparable performance to PLC, despite its simplicity.

3.7.4 Empirical Analysis

Correction Accuracy. The key idea of SELC is to correct the original noisy labels. We analyze the quality of the new target $\boldsymbol{t}$ by calculating its correction accuracy: $\frac{1}{N}\sum_i^N 1\{\arg\max \boldsymbol{y}_i = \arg\max \boldsymbol{t}_i\}$, where $\boldsymbol{y}_i$ is the true label of $\boldsymbol{x}_i$. Figure 3.4 (a) shows the correction accuracy of the Option I and Option II (SELC). We observe SELC achieves higher accuracy than Option I and correction accuracy is stable with the increase of training epochs. Figure 3.5 shows the confusion matrix of corrected labels w.r.t the true labels on CIFAR-10 with 40% symmetric label noise. SELC corrects the noisy labels impressively well for all classes.



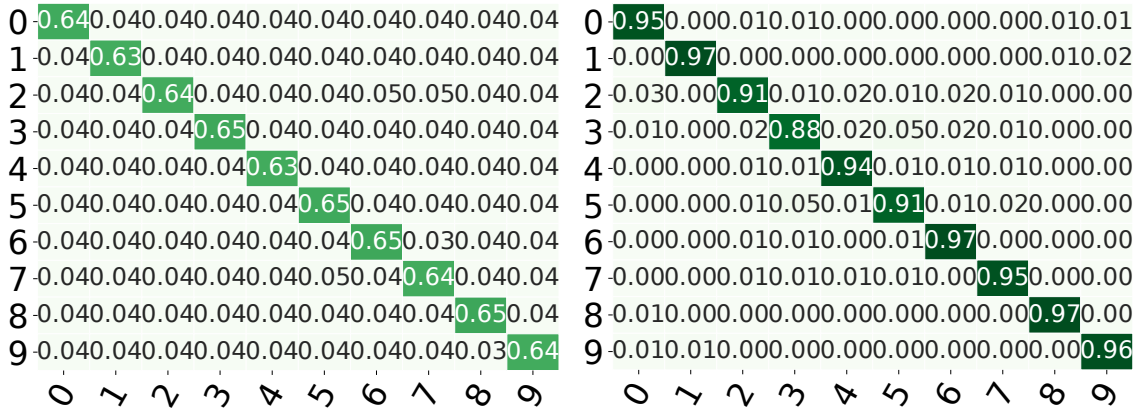
(a) Correction accuracy (b) Sensitivity of α

Figure 3.4: (a). Correction accuracy of *Option I* and *Option II* on CIFAR-10 with 40% (solid lines) and 80% (doted lines) symmetric noise. (b). Sensitivity of α.

Hyperparameter Sensitivity. In SELC, we have one hyperparameter α in Eq. (3.6) to control the momentum in self-ensemble. Figure 3.4 (b) shows the test accuracy when using different $\alpha \in \{0.85, 0.9, 0.95\}$ on CIFAR-10. We observe that the sensitivity to hyperparameter α is quite mild.

3.8 Conclusion

We propose a simple and effective method SELC to improve learning with noisy labels. SELC leverages the model predictions of previous epochs to correct the noisy labels, thus preventing the model from overfitting to noisy labels. By evaluating SELC on different types of label noise, we observe its superior performance over existing approaches.



(a) Before using SELC (b) After using SELC

Figure 3.5: (a). Confusion matrix of original noisy label w.r.t true labels on CIFAR-10 with 40% symmetric label noise. (b). Confusion matrix of corrected labels w.r.t true labels after using SELC. The numbers from 0 to 9 represent different classes in CIFAR-10 (e.g. 0 → airplane, 1 → automobile, . . . , 9 → truck).

3.9 Appendix

3.9.1 Training Details on Class-conditional Label Noise.

Preprocessing. We apply the standard data augmentation on CIFAR-10/100: horizontal random flip and 32 × 32 random crop after padding 4 pixels around images. The standard normalization with mean=(0.4914, 0.4822, 0.4465), std=(0.2023, 0.1994, 0.2010) is applied before feeding images to the network.

We conduct the experiments with class-conditional label noise on CIFAR-10 and CIFAR-100 (Krizhevsky *et al.*, 2009). Given these two datasets are initially clean, we follow (Patrini *et al.*, 2017) to inject noise by label transition matrix $\mathbf{Q}$, where $\mathbf{Q}_{ij} = \Pr[\hat{y} = j \mid y = i]$ denotes the probability that noisy label $\hat{y}$ is flipped from clean label y. We evaluate SELC in two types of noise: *symmetric* and *asymmetric*. Symmetric noise is generated by randomly replacing the labels for a percentage of

the training data with all possible labels. Asymmetric noise is designed to mimic the structure of real-world label noise, where the annotators are more likely to make mistakes only within very similar classes (e.g. *deer* $\rightarrow$ *horse* and *cat* $\leftrightarrow$ *dog*). We use the ResNet34 as backbone for both datasets, and train the model using SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 128. The network is trained for 200 epochs. We set the initial learning rate as 0.02, and reduce it by a factor of 10 after 40 and 80 epochs. For parameter α in SELC, we fix $\alpha = 0.9$. Note that we do not perform early stopping since we don't assume the presence of clean validation data. All test accuracy are recorded from the last epoch of training. The illustration of SELC+ are shown in Figure 3.6.

3.9.2 Training Details on Instance-dependent Label Noise.

For instance-dependent noise from SEAL, we use the same network architecture Wide ResNet28×10 (Zagoruyko and Komodakis, 2016). Models are trained for 150 epochs with a batch size of 128 and we report the test accuracy at the last epoch. We use SGD with a momentum of 0.9 and a weight decay of 5×10^{-4}. We use SGD with a momentum of 0.9 and a weight decay of 5×10^{-4}. The learning rate is initialized as 0.1 and is divided by 5 after 60 and 120 epochs. We report the test accuracy at the last epoch.

As for PMD noise, we evaluate SELC on three types of label noise from the PMD noise family for consistency with PLC. We use the same network architecture PreAct ResNet34 (He *et al.*, 2016b) as PLC. We use SGD with a momentum of 0.9 and a weight decay of 5×10^{-4}. For noise on CIFAR-10, models are trained for 150 epochs with a batch size of 128. The learning rate is initialized as 0.1 and is divided by 5

after 60 and 120 epochs. For noise on CIFAR-100, models are trained for 200 epochs with a batch size of 128. The learning rate is initialized as 0.02 and is divided by 10 after 40 and 80 epochs.

3.9.3 Training Details on Real-world Label Noise

Preprocessing. Following (Zhang *et al.*, 2020), for ANIMAL-10N, we apply normalization and regular data augmentation (i.e. horizontal flip) on the training sets. The standard normalization with mean=(0.485, 0.456, 0.406), std=(0.229, 0.224, 0.225) is applied before feeding images to the network. For Clothing1M and Webvision, we apply normalization and regular data augmentation (i.e. random crop and horizontal flip) on the training sets. The cropping size is consistent with existing works (Liu *et al.*, 2020). Specifically, 224×224 for Clothing 1M (after resizing to 256×256), and 227×227 for Webvision. The standard normalization with mean=(0.6959, 0.6537, 0.6371), std=(0.3113, 0.3192, 0.3214) for Clothing1M and mean=(0.485, 0.456, 0.406), std=(0.229, 0.224, 0.225) for Webvision.

ANIMAL-10N contains 50,000 human-labeled online images for 10 animals with confusing appearance. The estimated label noise rate is 8%. There are 50,000 training and 5,000 testing images. Following (Song *et al.*, 2019), we use VGG-19 with batch normalization and train it using SGD with a weight decay 0.001, batch size 128 and initial learning rate 0.1, which is reduced by a factor of 5 after 50 and 75 epochs (100 epochs in total). We repeat the experiments with 3 random trials and report the mean value and standard deviation. All test accuracy are recorded from the last epoch of training.

Clothing1M consists of 1 million training images collected from online shopping

websites. The labels are generated by using the surrounding texts of the images that are provided by the sellers and thus contain many wrong labels. The estimated label noise rate is 38.5%. It also contains 50k, 14k, and 10k of clean data for training, validation, and testing, respectively. Note that we do not use the 50k clean data in our training process. We use ResNet50 (He *et al.*, 2016a) pretrained on ImageNet and train it using SGD with a momentum 0.9, weight decay 0.001, batch size 64 and initial learning rate 0.01, which is reduced by a factor of 10 after 10 and 20 epochs (30 epochs in total). For each epoch, we sample 2000 mini-batches from the training data ensuring that the classses of the noisy labels are balanced. We report the accuracy on the test set when the performance on the validation set is optimal.

Webvision contains 2.4 million images crawled from the web using the 1,000 concepts in ImageNet ILSVRC12. The estimated label noise rate is 20%. Following (Chen *et al.*, 2019), we use the first 50 classes of Google image subset for training and test on the corresponding 50 classes of WebVision (approximate 66K) and ILSVRC-2012 validation set. We use InceptionResNetV2 (Szegedy *et al.*, 2016) and train it using SGD with a momentum 0.9, weight decay 0.0005, batch size 32 and initial learning rate 0.01, which is reduced by a factor of 10 after 40 and 80 epochs (100 epochs in total). We report the accuracy on the Webvision validation set and ImageNet ILSVRC-2012 validation set.

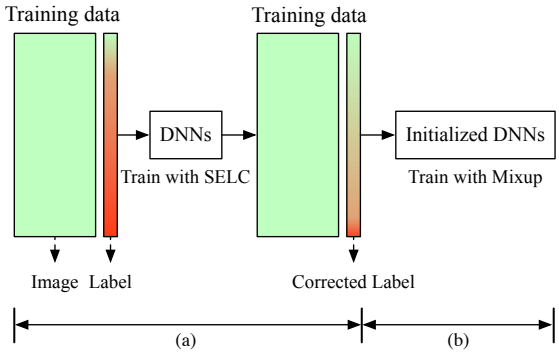


Figure 3.6: (a): We train the DNNs with SELC and get the corrected label $\boldsymbol{t}$. (b): We train an initialized DNNs with mixup using the corrected labels from (a).

Chapter 4

Noise Attention Learning

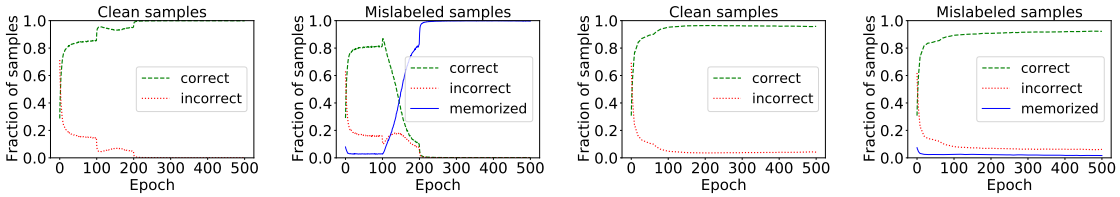
4.1 Citation and Main Contributor

Lu, Yangdi, Bo, Yang and He, Wenbo. "Noise Attention Learning." Accepted to NeurIPS 2022.

The main contributor to this paper is the first author - Yangdi Lu (contributes more than 80%).

4.2 Introduction

With the emergence of highly-curated datasets such as ImageNet (Deng *et al.*, 2009) and CIFAR (Krizhevsky *et al.*, 2009), deep neural networks (DNNs) have achieved remarkable performance on many classification tasks. However, in real-world applications, it is extremely time-consuming and expensive, sometimes even impossible to label a new large-scale dataset containing fully correct annotations. To alleviate this problem, one may obtain the data with lower quality annotations efficiently through



(a) CE (b) CE (c) NAL (ours) (d) NAL (ours)

Figure 4.1: The results of training a ResNet34 He *et al.* (2016b) with cross entropy (CE) loss and our proposed method on the CIFAR-10 with 40% symmetric noise. Plots (a) and (c) show the fraction of clean samples that are predicted correctly (green) and incorrectly (red). In contrast, plots (b) and (d) show the fraction of mislabeled samples that are predicted correctly (green), memorized (i.e. the prediction equals the wrong label, shown in blue), and incorrectly predicted as neither the true nor the labeled class (red). For clean samples, both models predict them correctly with the increasing of epochs. However, for mislabeled samples in plot (b), the model trained with CE loss first predicts the true labels correctly, but eventually memorizes all mislabeled samples. In contrast, our approach prevents memorization of mislabeled samples, allowing the model to continue learning the clean samples to attain high accuracy on all samples.

online keywords queries (Li *et al.*, 2017) or crowdsourcing (Yu *et al.*, 2018), but *noisy labels* (e.g. a cat image is mislabeled as dog) are inevitably introduced consequently. Previous studies (Zhang *et al.*, 2018a) demonstrate that noisy labels in training data are problematic for deep neural networks, resulting in overfitting and performance degradation. Therefore, designing robust algorithms against noisy labels is of great practical importance.

Given the training data consists of clean (correctly labeled) samples and mislabeled samples, DNNs have been observed to correctly predict the true labels for all training samples during a *early learning* stage, and then start to make incorrect predictions as it gradually *memorizes* the mislabeled samples (Arpit *et al.*, 2017; Liu *et al.*, 2020) (see Figure 4.1(a) and 4.1(b)). Based on this empirical finding, many existing approaches (Han *et al.*, 2018; Arazo *et al.*, 2019; Li *et al.*, 2020a; Pleiss *et al.*,

2020) use the training loss or prediction confidence to identify mislabeled samples in the early learning stage, i.e., the samples with small loss or high confidence are more likely to be clean samples. After dividing the samples into clean and mislabeled sets, one can train the model only based on clean set (Han *et al.*, 2018; Pleiss *et al.*, 2020), or correct these mislabeled samples by pseudo-labels (Arazo *et al.*, 2019), or treat them as unlabeled samples and train the model using semi-supervised learning algorithms (Li *et al.*, 2020a). However, these methods involve sophisticated training procedure modifications and sometimes require high computational costs.

In this paper, we propose Noise Attention Learning (NAL) which enjoys simplicity and universality from a new perspective. The core idea behind NAL is to let the model automatically differentiate the mislabeled samples from clean samples, and prevent the memorization of mislabeled samples in training. Since the memorization of DNNs has a preference for clean samples, the predictive power of a sample's representation aligns with its label cleanliness (Arpit *et al.*, 2017). To transform this qualitative observation into a quantitive measure, we introduce an attention branch that outputs scalar attention weights to indicate the predictive power of representations. We incorporate the attention weights in our loss function, so that the attention branch can be learned to divide samples, i.e., clean samples are learned to be associated with large weights, while mislabeled ones with small weights. In this way, the proposed method does not require extra steps (e.g. modeling loss distribution (Arazo *et al.*, 2019; Li *et al.*, 2020a), selecting small-loss samples (Han *et al.*, 2018)) to separate the training set, effectively simplifying the training procedure. Theoretically, we demonstrate that NAL neutralizes the effect of the mislabeled samples on the gradient and ensures the contribution of clean samples to the whole gradient remains dominant,

thereby preventing the deep neural networks from overfitting mislabeled samples.

The proposed method has a similar effect on gradients as a regularization method ELR (Liu *et al.*, 2020). In general, the performance of pure regularization methods cannot be compared to complex methods. Despite that the regularization methods can effectively hinder the memorization of mislabeled samples, the limited number of clean samples makes the model not rich enough to generalize effectively to the held-out data. To solve this problem, ELR uses temporal ensembling (Laine and Aila, 2016) to estimate the targets in its loss function to generate more clean labels for improving performance. Similarly, instead of directly using noisy labels as target distribution in our loss function, we adopt SELC (Lu and He, 2022) to estimate the targets, allowing the generation of more clean labels for learning sufficiency.

Empirically, we show that the proposed approach achieves better robustness than the state-of-the-art methods on two benchmarks (CIFAR-10/100 (Krizhevsky *et al.*, 2009)) with simulated label noise and three challenging datasets in noisy label literature (ANIMAL-10N (Song *et al.*, 2019), Clothing1M (Xiao *et al.*, 2015) and Webvision (Li *et al.*, 2017)) with real-world label noise. To better understand our method, we conduct many empirical analyses, including memorization analysis, gradient analysis, attention weights, quality of targets, hyperparameter sensitivity and ablation study, to verify our theoretical explanations and design goals.

4.3 Preliminaries

Classification with Noisy Labels. Consider the K-class classification problem in noisy-label scenario, the ground truth label y is unobservable. We only have a noisy training set $\hat{D} = \{(\boldsymbol{x}^{[i]}, \hat{y}^{[i]})\}_{i=1}^{N}$, where $\boldsymbol{x}^{[i]}$ is an input and $\hat{y}^{[i]} \in \mathcal{Y} = \{1, \ldots, K\}$ is

the corresponding noisy label. We denote $\hat{\boldsymbol{y}}^{[i]} \in \{0,1\}^K$ as one-hot vector of noisy label $\hat{y}^{[i]}$. A DNN $\mathcal{N}_\theta$ maps an input $\boldsymbol{x}^{[i]}$ to a K-dimensional logits and then feeds the logits to a softmax function to obtain $\boldsymbol{p}^{[i]}$ of the conditional probability of each class. θ denotes the parameters of the DNN and $\boldsymbol{z}^{[i]} \in \mathbb{R}^{K\times 1}$ denotes the logits (i.e. pre-softmax output). We refer to it as prediction branch in this work. We have $\boldsymbol{z}^{[i]} = \mathcal{N}_\theta(\boldsymbol{x}^{[i]})$ and $\boldsymbol{p}^{[i]} = \texttt{softmax}(\boldsymbol{z}^{[i]})$. Without knowing the ground truth joint probability distribution $P(\boldsymbol{x}, y)$, the cross-entropy (CE) loss is often used as empirical risk to measure how well the model fits the training set $\hat{D}$ as follows:

$$\mathcal{L}_{\text{ce}} = \frac{1}{N}\sum_{i=1}^{N} \ell_{\text{ce}}(\hat{\boldsymbol{y}}^{[i]}, \boldsymbol{p}^{[i]}) = -\frac{1}{N}\sum_{i=1}^{N} (\hat{\boldsymbol{y}}^{[i]})^\top \log(\boldsymbol{p}^{[i]}). \tag{4.1}$$

Early Learning Phenomenon. When optimizing $\mathcal{L}_{\text{ce}}$ by stochastic gradient descent (SGD), it has been observed that DNNs fit the easy (clean) samples first before memorizing of hard (mislabeled) samples (Arpit *et al.*, 2017). Figure 4.1(a) and 4.1(b) shows this qualitative behavior. The model begins by learning to predict the true labels, even for majority of mislabeled samples, but eventually learns to predicts wrong labels due to memorization of mislabeled samples.

Label Noise Model. The generation of real-world label noise is unpredictable, a typical methodology to dealing with noisy labels is to posit a noise model and develop robust algorithms under this model. The algorithms are then tested on the real-world datasets to see how effective they are. A widely-accepted noise model is class-conditional noise (Natarajan *et al.*, 2013; Patrini *et al.*, 2017), where label noise is independent of input features and true labels are corrupted by a *symmetric* or *asymmetric* noise transition matrix (simulation details in Section 4.6). Instance-dependent noise (Cheng *et al.*, 2020) is another noise model in which the noise is

affected not only by the label but also by the input feature.

4.4 Noise Attention Learning

In this section, we present our approach NAL for learning with noisy labels. NAL consists of three key elements: (1) An attention branch based on learned representations to produce the attention weight for each training sample; (2) A noise attention loss is specifically designed to learn the attention weights; (3) A target estimation strategy to help generate more clean labels for learning.

4.4.1 Attention Branch based on Learned Representations

To enable the deep neural networks to produce an attention weight that reflects the quality of learned representation, we introduce an attention branch just after the penultimate layer of the original model. We denote M-dimensional penultimate layer (representation) of input $\boldsymbol{x}^{[i]}$as $H^{[i]}$. $H^{[i]}$ is shared in both prediction and attention branches. For each input, the prediction branch outputs the softmax prediction $\boldsymbol{p}^{[i]}$ as usual. The attention branch contains one fully connected layer to produce a single scalar value $h^{[i]}$, and `sigmoid` function is applied to scale it between 0 to 1. Specifically, we have $h^{[i]} = WH^{[i]} + b$, where $W \in \mathbb{R}^{1\times M}$ denotes the weights and $b \in \mathbb{R}$ denotes the bias in penultimate layer of the attention branch. We have the attention weight $tau^{[i]}$ for each sample $\boldsymbol{x}^{[i]}$ as

$$\tau^{[i]} = \texttt{sigmoid}(h^{[i]}), \quad \tau^{[i]} \in (0, 1). \tag{4.2}$$

Note that W and b are extra network parameters need to be learned to produce meaningful τ's. Therefore, attention weights are not fixed but dynamic during the training.

4.4.2 Noise Attention Loss

The early learning phenomenon reveals that the DNNs memorize the clean samples before the mislabeled samples. Therefore, clean samples are likely to have better learned representations than mislabeled samples in the early learning phase. To enable the attention weights to automatically capture the difference of representations as well as learn the classification task, we propose a noise attention loss consisting of two terms: attention term $\mathcal{L}_{\mathrm{a}}$ and boost term $\mathcal{L}_{\mathrm{b}}$.

$$\mathcal{L}_{\mathrm{NAL}} = \mathcal{L}_{\mathrm{a}} + \lambda \mathcal{L}_{\mathrm{b}} = -\frac{1}{N}\sum_{i=1}^{N} (\hat{\boldsymbol{y}}^{[i]})^{\top} \log\big(\underbrace{\tau^{[i]}(\boldsymbol{p}^{[i]} - \hat{\boldsymbol{y}}^{[i]}) + \hat{\boldsymbol{y}}^{[i]}}_{\text{perceptual shortcut}}\big) - \frac{\lambda}{N}\sum_{i=1}^{N} \log(\tau^{[i]}), \tag{4.3}$$

where λ is the hyperparameter. Intuitively, the attention term $\mathcal{L}_{\mathrm{a}}$ can be interpreted as a dynamic expansion of the CE loss, where the prediction is modified to 'perceptual shortcut' $\tau(\boldsymbol{p} - \hat{\boldsymbol{y}}) + \hat{\boldsymbol{y}}$ (Note that many existing methods (Reed *et al.*, 2015; Ma *et al.*, 2018; Arazo *et al.*, 2019) use this linear interpolation formula to infer the label distribution, while the proposed NAL applies it to prediction distribution for learning the meaningful attention weights). Assuming no representations are corrupted and all weights $\tau \to 1$, the attention term $\mathcal{L}_{\mathrm{a}}$ reduces to exactly $\mathcal{L}_{\mathrm{ce}}$ in Eq. (3.1). On the contrary, when all $\tau \to 0$, the attention term $\mathcal{L}_{\mathrm{a}}$ becomes "lazy" as it produces many zero gradients.

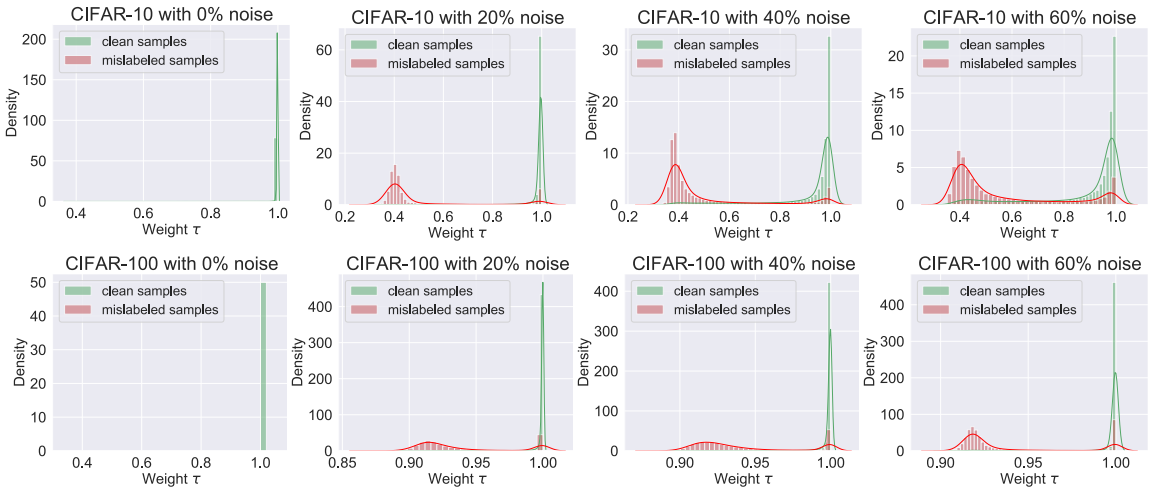


Figure 4.2: The distribution of attention weights on the CIFAR-10/CIFAR-100 with different ratios of symmetric label noise using ResNet34.

The mechanism of attention term can be explained as follows: (1) For mislabeled samples, the model does not memorize them in the early learning phase. Therefore, their representations are under learned compared to clean samples and $(\boldsymbol{p}-\hat{\boldsymbol{y}})$ remains large. By minimizing $\mathcal{L}_{\mathrm{a}}$, it forces τ of mislabeled samples toward 0, which is treated as 'shortcut'. (2) For clean samples, the model memorizes them first, resulting in $(\boldsymbol{p}-\hat{\boldsymbol{y}}) \rightarrow \mathbf{0}$. It makes τ have no influence on minimizing $\mathcal{L}_{\mathrm{a}}$ as the prediction distribution is already equal to $\hat{\boldsymbol{y}}$. As a result, by solely minimizing $\mathcal{L}_{\mathrm{a}}$, we may obtain a trivial optimization result that the model always produces $\tau \rightarrow 0$ for any inputs. To avoid this 'lazy' learning circumstance, we have another boost term $\mathcal{L}_{\mathrm{b}}$. It can be interpreted as a binary CE loss, where the target of τ is always 1 for all inputs. By adding the boost term, the attention weights of clean samples are pushed to 1, and weights of mislabeled samples are still close to 0 as expected.

To verify the effectiveness of $\mathcal{L}_{\mathrm{NAL}}$ in discriminating the mislabeled samples from

clean samples, we empirically analyze the weight distribution of clean and mislabeled samples with simulated symmetric noise. Figure 4.2 shows results on CIFAR-10 and CIFAR-100 with different ratios of symmetric noise. In 0% noise cases, we observe that all samples have weights close to 1, which indicates that $\mathcal{L}_{\mathrm{NAL}}$ reduced to exactly $\mathcal{L}_{\mathrm{ce}}$. Thus, the proposed loss would not affect the performance when training with clean data. At other ratios of label noise, there is a clear separation between attention weights of clean and mislabeled samples. The weights of clean samples form a spike close to 1.0, whereas the weights of most mislabeled samples are plainly less than 1.0, which satisfies our design goal.

4.4.3 Target Estimation

For regularization methods, directly using the noisy labels as target distribution in loss function is less effective as the limited number of clean samples makes the model not rich enough to generalize effectively to the held-out data, especially at the high noise level. To yield better performance, ELR (Liu *et al.*, 2020) use the semi-supervised technique temporal ensembling (Laine and Aila, 2016) to estimate targets in its regularizer and steer the model towards these targets. The ensemble predictions are formed by an exponential moving average of network outputs. Specifically, the ensemble predictions in the m-th epoch can be expressed as $\tilde{\boldsymbol{p}}_{[m]} = \sum_{j=1}^{m}(1-\alpha)\alpha^{m-j}\boldsymbol{p}_{[j]}$, where $0 \leq \alpha < 1$ is the momentum. In this paper, we use the ensemble strategy in SELC (Lu and He, 2022) to estimate our targets $\boldsymbol{t}$ for the m-th epoch as follows:

$$\boldsymbol{t}_{[m]} = \underbrace{\alpha^m \hat{\boldsymbol{y}}}_{\text{initial term}} + \underbrace{\sum_{j=1}^{m}(1-\alpha)\alpha^{m-j}\boldsymbol{p}_{[j]}}_{\text{ensemble term}}. \tag{4.4}$$

Algorithm 3: Noise Attention Learning (NAL) pseudocode

Input: DNN $\mathcal{N}_\theta$, hyperparameters λ and α, training set $\hat{D} = \{(\boldsymbol{x}^{[i]}, \hat{y}^{[i]})\}_{i=1}^{N}$, total epoch E, mini batch B (Note that the hyperparameters for each different datasets can be found in Appendix 4.9.3);

for *epoch* $e = 1$ ***to*** E **do**
  for *each minibatch* B **do**
    for i *in* B **do**
      Obtain $\boldsymbol{z}^{[i]}, h^{[i]} = \mathcal{N}_\theta(\boldsymbol{x}^{[i]})$;
      Obtain $\boldsymbol{p}^{[i]}, \tau^{[i]} = \texttt{softmax}(\boldsymbol{z}^{[i]}), \texttt{sigmoid}(h^{[i]})$;
      Update $\boldsymbol{t}_{[e]}^{[i]}$ by Eq. (4.4);
  Calculate
  $\mathcal{L}_{\text{NAL}} = -\frac{1}{|B|}\sum_{i=1}^{|B|}(\boldsymbol{t}_{[e]}^{[i]})^\top \log\left(\tau^{[i]}(\boldsymbol{p}^{[i]} - \boldsymbol{t}_{[e]}^{[i]}) + \boldsymbol{t}_{[e]}^{[i]}\right) - \frac{\lambda}{|B|}\sum_{i=1}^{|B|}\log(\tau^{[i]})$.;
  Update θ using stochastic gradient descent;
Output: θ.

The initial term preserves the original noisy labels with exponential decaying weights α^m. In the early learning phase (small m), the overall target relies on noisy labels to learn the model. As the learning continues (larger m), α^m approaches 0. The overall target eventually depends on the ensemble term $\tilde{\boldsymbol{p}}_{[m]}$ and penalizes model predictions that are inconsistent with this target. In our experiment, we will demonstrate that estimated targets are more accurate than original noisy labels. We simply replace noisy label $\hat{\boldsymbol{y}}$ with ensemble target $\boldsymbol{t}$ in our loss. Algorithm 3 shows pseudocode of our method.

4.5 Theoretical Justification

In this section, we demonstrate the noise robustness of $\mathcal{L}_{\text{NAL}}$ by analyzing how it scales the gradients accordingly to prevent memorization of mislabeled samples. The omitted proofs in this section are in appendix. For clarity of explanation, we denote

the true label of sample $\boldsymbol{x}$ as $y \in \{1, ..., K\}$. The ground-truth distribution over labels for sample $\boldsymbol{x}$ is $q(y|\boldsymbol{x})$, and $\sum_{k=1}^{K} q(k|\boldsymbol{x}) = 1$. Consider the case of a single ground-truth label y, then $q(y|\boldsymbol{x}) = 1$ and $q(k|\boldsymbol{x}) = 0$ for all $k \neq y$. We denote the prediction probability as $p(k|\boldsymbol{x})$ and $\sum_{k=1}^{K} p(k|\boldsymbol{x}) = 1$. For notation simplicity, we denote p_k, q_k, p_y, q_y, p_j, q_j as abbreviations for $p(k|\boldsymbol{x})$, $q(k|\boldsymbol{x})$, $p(y|\boldsymbol{x})$, $q(y|\boldsymbol{x})$, $p(j|\boldsymbol{x})$ and $q(j|\boldsymbol{x})$. First, we explain how the CE loss fails in noisy-label scenario.

Lemma 4.5.1. *Given the cross-entropy loss $\mathcal{L}_{ce}$ in Eq. (4.1), we rewrite the sample-wise loss $\ell_{ce} = -\sum_{k=1}^{K} q_k \log p_k$. Its gradient with respect to z_j is*

$$\frac{\partial \ell_{ce}}{\partial z_j} = \begin{cases} p_j - 1 \leq 0, & q_j = q_y = 1, \\ \\ p_j \geq 0, & q_j = 0, \end{cases} \tag{4.5}$$

where z_j is the j-th entry of logits $\boldsymbol{z}$. $q_j = q_y = 1$ means j equals the true class y.

In clean-label scenario, Lemma 4.5.1 ensures that, during SGD, learning direction of CE loss continues towards the true class as the corresponding gradient remains negative. However, in noisy-label scenario, the learning direction fluctuates. Suppose j is true class and equals y, but $q_j = 0$ due to the label noise, then the contribution of a mislabeled sample to the gradient is reversed (i.e. gradient should be negative but get positive instead). The entry corresponding to the impostor class j', is also reversed because $q_{j'} = 1$, causing the gradient of mislabeled samples dominates (in Figure 4.3(a)). Consequently, using CE loss results in memorization of the mislabeled samples.

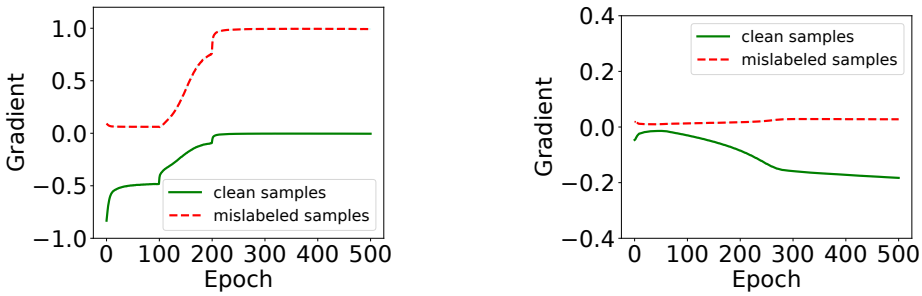
Theorem 4.5.1. *Given the noise attention loss $\mathcal{L}_{NAL}$ in Eq. (4.3), we rewrite the sample-wise loss $\ell_{NAL} = -\sum_{k=1}^{K} q_k \log(\tau(p_k - q_k) + q_k) - \lambda \log \tau$. Its gradient with*

respect to the logits z_j can be derived as

$$\frac{\partial \ell_{NAL}}{\partial z_j} = \begin{cases} \frac{p_y}{p_y - 1 + 1/\tau}(p_j - 1) \leq 0, & q_j = q_y = 1 \ (j \textit{ is the true class for } \boldsymbol{x}) \quad (4.6a) \\ \frac{p_y}{p_y - 1 + 1/\tau} p_j \geq 0, & q_j = 0 \ (j \textit{ is not the true class for } \boldsymbol{x}) \quad (4.6b) \end{cases}$$

where the multiplier $\frac{p_y}{p_y - 1 + 1/\tau} \in (0, 1)$.

Theorem 4.5.1 ensures that learning on true class persists when training with noise attention loss. In addition, compared to the gradient of ℓ_{ce}, the gradient of ℓ_{NAL} is scaled by a positive multiplier term $\frac{p_y}{p_y - 1 + 1/\tau}$. Let's denote $\varphi = \frac{p_y}{p_y - 1 + 1/\tau}$. φ is monotonically increasing on τ. We have $\lim_{\tau \to 1} \varphi = 1$, and $\lim_{\tau \to 0} \varphi = 0$. For the samples with the true class j in Eq. (4.15a), the cross-entropy gradient term $p_j - 1$ of clean samples tends to vanish after early learning stage, causing mislabeled samples to dominate the gradient. However, by multiplying φ (note that $\varphi \to 0$ for mislabeled samples and $\varphi \to 1$ for clean samples due to attention weight distribution), it counteracts the effect of gradient dominating by mislabeled samples. For the samples that j is not the true class in Eq. (4.15b), the gradient term p_j is positive. Multiplying $\varphi < 1$ effectively dampens the magnitudes of coefficients on these mislabeled samples, thereby diminishing their effect on the gradient. Figure 4.3(b) empirically shows the gradient when training with $\mathcal{L}_{\text{NAL}}$. NAL keeps the gradient of clean samples dominant and diminishes the gradient of mislabeled samples when epoch increases, effectively preventing memorization of mislabeled samples.



(a) Gradient of $\mathcal{L}_{\text{ce}}$ (b) Gradient of $\mathcal{L}_{\text{NAL}}$

Figure 4.3: The results of gradient on CIFAR-10 with 40% symmetric label noise using ResNet34. We observe that in plot (a), the gradient of clean samples dominates in early learning stage, but afterwards it vanishes (close to 0) and the gradient of mislabeled samples dominates. In plot (b), NAL effectively keeps the gradient of clean samples dominant and diminishes the gradient of mislabeled samples.

4.6 Experiments

In this section, we first compare NAL with the existing methods on five widely used datasets. Then we provide several empirical results towards a better understanding of NAL. Finally, we provide an ablation study to evaluate the influence of each component and also show the results when integrated with another noise robust method SCE (Wang *et al.*, 2019). All experiments are implemented in PyTorch and run in a single NIVDIA A100 GPU.

4.6.1 Comparison with Existing Methods

Datasets and Setups. We evaluate our approach on two benchmarks CIFAR-10 and CIFAR-100 (Krizhevsky *et al.*, 2009) with simulated label noise, and three real-world datasets, ANIMAL-10N (Song *et al.*, 2019), Clothing1M (Xiao *et al.*, 2015) and WebVision (Li *et al.*, 2017). Since CIFAR-10 and CIFAR-100 are initially clean, we follow (Patrini *et al.*, 2017) to inject symmetric and asymmetric label noises.

Symmetric noise is generated by uniformly flipping the label to one of the other class label. Asymmetric noise is a simulation of fine-grained classification, where the label flipping only occurs within very similar classes (e.g. dog $\leftrightarrow$ cat). ANIMAL-10N contains human-labeled online images for 10 animals with confusing appearance. Its estimated noise rate is 8%. Clothing1M consists of 1 million images collected from online shopping websites with labels generated from surrounding texts. Its estimated noise rate is 38.5%. WebVision contains 2.4 million images crawled from the web using the 1000 concepts in ImageNet ILSVRC12. Its estimated noise rate is 20%. For CIFAR-10/100, We use a ResNet-34 (He *et al.*, 2016a) and train it using SGD with a batch size of 64. For ANIMAL-10N, we use VGG-19 (Simonyan and Zisserman, 2015) with batch normalization and train it using SGD with a batch size of 128. For Clothing1M, we train a ResNet-50 (He *et al.*, 2016a) using SGD with a batch size of 64. For Webvision, we train a InceptionResNetV2 (Szegedy *et al.*, 2016) using SGD with a batch size of 32. More training details can be found in appendix.

Dataset	CIFAR-10					CIFAR-100				
Noise type	symm				asymm	symm				asymm
Noise ratio	20%	40%	60%	80%	40%	20%	40%	60%	80%	40%
Cross Entropy	86.98 ± 0.12	81.88 ± 0.29	74.14 ± 0.56	53.82 ± 1.04	80.11 ± 1.44	58.72 ± 0.26	48.20 ± 0.65	37.41 ± 0.94	18.10 ± 0.82	42.74 ± 0.61
Forward $\hat{T}$ (Patrini *et al.*, 2017)	87.99 ± 0.36	83.25 ± 0.38	74.96 ± 0.65	54.64 ± 0.44	83.55 ± 0.58	39.19 ± 2.61	31.05 ± 1.44	19.12 ± 1.95	8.99 ± 0.58	34.44 ± 1.93
Bootstrap (Reed *et al.*, 2015)	86.23 ± 0.23	82.23 ± 0.37	75.12 ± 0.56	54.12 ± 1.32	81.21 ± 1.47	58.27 ± 0.21	47.66 ± 0.55	34.68 ± 1.10	21.64 ± 0.97	45.12 ± 0.57
GCE (Zhang and Sabuncu, 2018)	89.83 ± 0.20	87.13 ± 0.22	82.54 ± 0.23	64.07 ± 1.38	76.74 ± 0.61	66.81 ± 0.42	61.77 ± 0.24	53.16 ± 0.78	29.16 ± 0.74	47.22 ± 1.15
Joint Opt (Tanaka *et al.*, 2018)	92.25	90.79	86.87	69.16	-	58.15	54.81	47.94	17.18	-
NLNL (Kim *et al.*, 2019)	94.23	92.43	88.32	-	89.86	71.52	66.39	56.51	-	45.70
SCE (Wang *et al.*, 2019)	89.83 ± 0.20	87.13 ± 0.26	82.81 ± 0.61	68.12 ± 0.81	82.51 ± 0.45	70.38 ± 0.13	62.27 ± 0.22	54.82 ± 0.57	25.91 ± 0.44	69.32 ± 0.87
DAC (Thulasidasan *et al.*, 2019a)	92.91	90.71	86.30	74.84	-	73.55	66.92	57.17	32.16	-
SAT† (Huang *et al.*, 2020)	94.14	92.64	89.23	78.58	-	75.77	71.38	62.69	**38.72**	-
SELC (Lu and He, 2022)	93.09 ± 0.02	91.18 ± 0.06	87.25 ± 0.09	74.13 ± 0.14	91.05 ± 0.11	73.63 ± 0.07	68.46 ± 0.10	59.41 ± 0.06	32.63 ± 0.06	70.82 ± 0.09
ELR† (Liu *et al.*, 2020)	92.12 ± 0.35	91.43 ± 0.21	88.87 ± 0.24	80.69 ± 0.57	90.35 ± 0.38	74.68 ± 0.31	68.43 ± 0.42	60.05 ± 0.78	30.27 ± 0.86	73.73 ± 0.34
NAL† (ours)	**94.37 ± 0.04**	**93.49 ± 0.07**	**90.56 ± 0.07**	**80.98 ± 0.27**	**92.09 ± 0.12**	**77.79 ± 0.28**	**74.65 ± 0.09**	**68.48 ± 0.16**	36.77 ± 0.71	**74.73 ± 0.12**

Table 4.1: Test Accuracy (%) on CIFAR-10 and CIFAR-100 with various levels of label noise injected to the training set. We compare with previous works under the same backbone ResNet34. The results are averaged over 3 trials. Results are taken from their original papers. The best results are in **bold**. The marker † denotes the results with cosine annealing learning rate Loshchilov and Hutter (2016).

Table 4.1 shows the results on CIFAR-10 and CIFAR-100 with different levels of

CE	Nested (Chen *et al.*, 2021c)	SELFIE (Song *et al.*, 2019)	PLC (Zhang *et al.*, 2020)	SELC (Lu and He, 2022)	Nested + Co-teaching (Chen *et al.*, 2021c)	NAL (ours)
79.40 ± 0.14	81.30 ± 0.60	81.80 ± 0.09	83.40 ± 0.43	83.73 ± 0.06	84.10 ± 0.10	**84.18 ± 0.19**

Table 4.2: The accuracy (%) results on ANIMAL-10N. All methods use a VGG-19 architecture. Results of other methods are taken from original papers.

CE	SCE (Wang *et al.*, 2019)	DMI (Xu *et al.*, 2019)	ODNL (Wei *et al.*, 2021a)	ELR (Liu *et al.*, 2020)	FINE (Kim *et al.*, 2021)	Nested (Chen *et al.*, 2021c)	HOC (Zhu *et al.*, 2021)	NAL‡ (ours)
69.21	71.02	72.46	72.47	72.87	72.91	73.10	73.39	**73.58**

Table 4.3: The accuracy (%) results on Clothing1M. All compared methods use the ResNet-50 pretrained on ImageNet. The marker ‡ denotes the model is trained from scratch.

symmetric and asymmetric label noise. We compare NAL to the best performing approaches that only modify the training loss. All of these methods use the same backbone (ResNet34). NAL shows substantial improvements over other methods and obtains the highest accuracy in most cases.

Table 4.2 shows the results on ANIMAL-10N. NAL achieves state-of-the-art performance, even better than Nested which uses Co-teaching to boost performance.

Table 4.3 shows the results on the Clothing1M. All compared methods are trained

		Co-teaching (Han *et al.*, 2018)	Iterative-CV (Chen *et al.*, 2019)	CRUST (Mirzasoleiman *et al.*, 2020)	ODD (Song *et al.*, 2020)	NCT (Sarfraz *et al.*, 2021)	ELR (Liu *et al.*, 2020)	NAL (ours)
WebVision	top1	63.58	65.24	72.40	74.60	75.16	76.26	**77.41**
	top5	85.20	85.34	89.56	90.60	90.77	91.26	**92.25**
ILSVRC12	top1	61.48	61.60	67.36	57.80	71.73	68.71	**74.09**
	top5	84.70	84.98	87.84	86.30	91.61	87.84	**92.09**

Table 4.4: The accuracy (%) results on (mini) Webvision. Results of other methods are taken from their original papers. All methods use an InceptionResNetV2 architecture.

using ResNet50 pretrained on ImageNet. However, as NAL needs to learn the parameters W and b in the attention branch, we directly train our model from scratch. As we can observe from Table 4.3, NAL still slightly outperforms other methods even without using an ImageNet pretrained model.

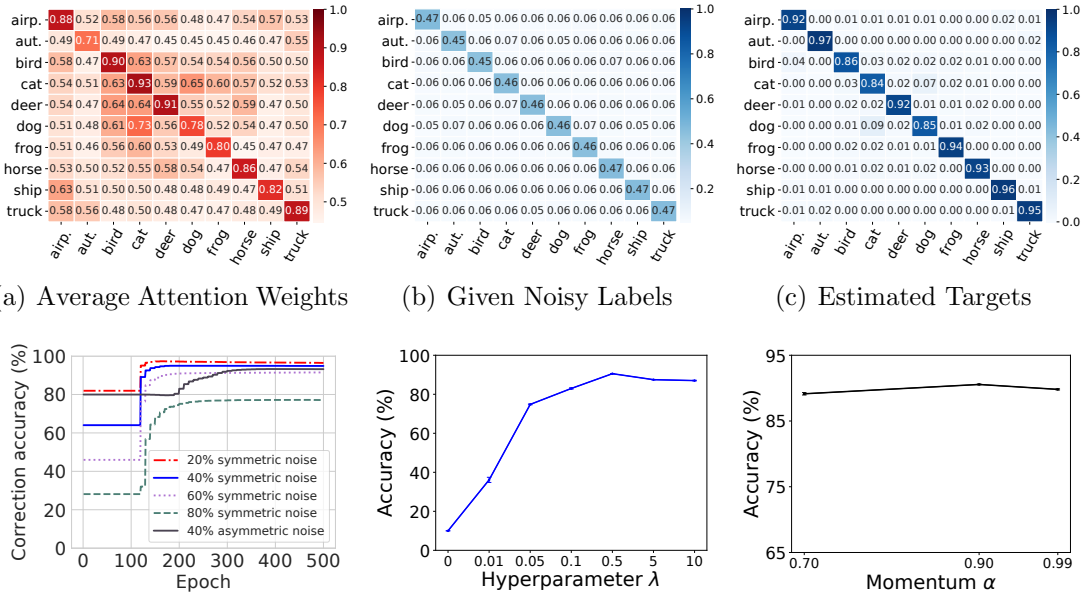
Table 4.4 shows the results on the (mini) WebVision following (Chen *et al.*, 2019). All the compared methods are evaluated on WebVision and ImageNet ILSVRC12 validation sets. We observe that NAL consistently outperforms state-of-the-art methods on both validation sets in terms of top1 and top5 accuracy.

Note that we do not compare with some state-of-the-art methods like RoCL (Zhou *et al.*, 2020) and DivideMix (Li *et al.*, 2020a) as baselines, because these methods are aggregations of multiple techniques (e.g. mixup (Zhang *et al.*, 2018b), multiple networks (Han *et al.*, 2018), and complex augmentation (Cubuk *et al.*, 2020)), while this paper only focuses on one, so the comparison seems unfair.

4.6.2 Empirical Analysis of NAL

Memorization Procedure of NAL. Figure 4.1(c) and 4.1(d) show the memorization procedure of NAL on clean and mislabeled samples, respectively. Compared to CE loss in Figure 4.1(a) and 4.1(b), NAL effectively prevents memorization of mislabeled samples, allowing the model to continue learning on the clean samples to attain high classification accuracy.

Gradient of $\mathcal{L}_{\mathbf{ce}}$ and $\mathcal{L}_{\mathbf{NAL}}$. Figure 4.3(a) and 4.3(b) show the gradients of CE loss and noise attention loss on CIFAR-10 with 40% symmetric noise using ResNet34 He *et al.* (2016a) respectively. In comparison to CE loss, the proposed method keeps



(a) Average Attention Weights (b) Given Noisy Labels (c) Estimated Targets

(d) Correction Accuracy (e) Sensitivity of λ (f) Sensitivity of α

Figure 4.4: (a). Average attention weights of noisy labels w.r.t clean labels on CIFAR-10 with 60% symmetric noise. (b). Confusion matrix of noisy labels w.r.t true labels on CIFAR-10 with 60% symmetric noise. (c). Confusion matrix of corrected labels w.r.t true labels on CIFAR-10 with 60% symmetric noise. (d). Label correction accuracy vs. epochs on CIFAR-10 with different levels of noise. Plots (e) and (f) show the sensitivity of λ and α on CIFAR-10 with 60% symmetric noise.

the gradient of clean samples dominant and dampens the gradient of mislabeled samples, resulting in the whole gradient being guided by clean samples.

Attention Weights. When trained on noisy labels, we have observed that the mislabeled samples have smaller weights than the clean samples in Figure 4.2. Here, we report the average weights of samples in Figure 4.4(a). The (i, j)-th block represents the average weights of samples with clean label i and noisy label j. We observe that the attention weights on the diagonal blocks are higher than those on non-diagonal blocks, which indicates NAL does properly down-weight the mislabeled samples.

Quality of Estimated Targets. We evaluate the quality of new targets $\boldsymbol{t}$ by

correction accuracy defined via $\frac{1}{N}\sum_i^N 1\{\text{argmax}_j\ \boldsymbol{y}_j^{[i]} = \text{argmax}_j\ \boldsymbol{t}_j^{[i]}\}$, where $\boldsymbol{y}^{[i]}$ is the true label of input $\boldsymbol{x}^{[i]}$. Figure 4.4(d) shows the correction accuracy vs. epochs on CIFAR-10 with different levels of label noise. The target estimation in Section 4.4.3 steadily improves the quality of new targets. Figure 4.4(b) and Figure 4.4(c) shows the confusion matrix of noisy labels and corrected labels (i.e. the hard version of new targets) w.r.t the clean labels on CIFAR-10 with 60% symmetric noise respectively. We observe that target estimation corrects the noisy labels impressively well for all classes.

Sensitivity of Hyperparameters λ and α. NAL contains two hyperparameters: λ is the coefficient for boost term $\mathcal{L}_{\text{b}}$, α is the momentum in target estimation. Figure 4.4(e) and 4.4(f) show their sensitivity to the performance. We observe that the sensitivity to α is quite mild as long as it is set close to 1. λ need to be tuned according to the complexity of dataset. It cannot be set to be very large or very small, resulting in neglecting attention term or falling into the 'lazy' learning circumstance. In our experiments, we fixed $\alpha = 0.9$ for all datasets. For CIFAR-10 and ANIMAL-10N, we set $\lambda = 0.5$. For CIFAR-100, we set $\lambda = 10$. For Webvision and Clothing1M, we set $\lambda = 50$.

4.6.3 Ablation Study

Table 4.5 reports the influence of the two components in NAL: target estimation and attention branch. Removing the target estimation leads to a significant performance drop. This suggests that using target estimation is crucial as it generates more clean samples for the model to learn. To validate the effect of the attention branch, we conduct another way to calculate the attention weights: using the confidence score

Dataset	CIFAR-10			CIFAR-100		
Noise type	symm		asymm	symm		asymm
Noise ratio	40%	80%	40%	40%	80%	40%
NAL	**93.49 ± 0.07**	**80.98 ± 0.27**	**92.09 ± 0.12**	74.65 ± 0.09	36.77 ± 0.71	74.73 ± 0.12
NAL w/o target estimation	89.47 ± 0.50	76.91 ± 0.22	88.23 ± 0.22	69.91 ± 0.21	31.33 ± 0.38	55.68 ± 0.17
NAL w/o attention branch	90.94 ± 0.28	⊖	91.55 ± 0.07	⊖	⊖	⊖
NAL + RNAL	92.92 ± 0.29	80.20 ± 0.42	88.17 ± 0.63	**75.38 ± 0.08**	**38.24 ± 0.55**	**74.89 ± 0.20**

Table 4.5: Ablation study results in terms of test accuracy (%) on CIFAR-10 and CIFAR-100. ⊖ means the model fails to converge. + RNAL means adding the reverse term of noise attention loss.

(highest probability in output) as the weight, i.e., $\max_j \boldsymbol{p}_j, j \in [1, K]$. We observe that the model does not converge in hard cases. We conjecture that using the confidence from model output does interfere with the original prediction branch, while adding the attention branch to get attention weight effectively solve this problem. We also explore the possibility of integrating NAL with other methods. Here we adopt the idea from symmetric cross entropy (SCE) (Wang *et al.*, 2019), which adds the reverse term to achieve noise robustness. As shown in Table 4.5, adding RNAL affects the performance under CIFAR-10, but enjoys performance boost under CIFAR-100 without extra cost.

4.7 Related Work

We briefly discuss the existing noise-robust methods that do not require a set of clean training data (as opposed to (Xiao *et al.*, 2015; Vahdat, 2017; Hendrycks *et al.*, 2018)). **Loss Correction.** Many approaches focus on correcting the loss function explicitly by estimating the noise transition matrix (Goldberger and Ben-Reuven, 2017; Patrini *et al.*, 2017; Tanno *et al.*, 2019). **Robust Loss Functions.** These

studies develop loss functions that are robust to label noise, including $\mathcal{L}_{\text{DMI}}$ (Xu *et al.*, 2019), MAE (Ghosh *et al.*, 2017), GCE (Zhang and Sabuncu, 2018), SCE (Wang *et al.*, 2019), NCE (Ma *et al.*, 2020), TCE (Feng *et al.*, 2021) and GJS (Englesson and Azizpour, 2021). Above two categories of methods do not utilize the early learning phenomenon. **Sample Selection** During the early learning stage, the samples with smaller loss values are more likely to be the clean samples. Based on this observation, MentorNet (Jiang *et al.*, 2018) pre-trains a mentor network for selecting small-loss samples to guide the training of the student network. Co-teaching related methods (Han *et al.*, 2018; Yu *et al.*, 2019; Wei *et al.*, 2020; Lu *et al.*, 2022) maintain two networks, and each network is trained on the small-loss samples selected by its peer network. **Label Correction** Joint Opt (Tanaka *et al.*, 2018) and PENCIL (Yi and Wu, 2019) replace the noisy labels with soft (i.e. model probability) or hard (i.e to one-hot vector) pseudo-labels. Arazo *et al.* (2019) weigh the clean and mislabeled samples by fitting a two-component Beta mixture model to loss values, and corrects the labels via mixup combination. Similarly, DivideMix (Li *et al.*, 2020a) trains two networks to separate the clean and mislabeled samples via a two-component Gaussian mixture model, and further uses MixMatch (Berthelot *et al.*, 2019) to enhance the performance. **Regularization** Li *et al.* (2020b) prove the gradient descent with early stopping is an effective regularization to achieve robustness to label noise. Hu *et al.* (2019) *explicitly* add the regularizer based on neural tangent kernel (Jacot *et al.*, 2018) to limit the distance between the model parameters to initialization. ELR (Liu *et al.*, 2020) estimates the target by temporal ensembling (Laine and Aila, 2016) and adds a regularization term to cross-entropy loss to avoid memorization. Other techniques, such as mixup augmentation (Zhang *et al.*, 2018b), label smoothing (Szegedy *et al.*,

2016) and weight averaging (Tarvainen and Valpola, 2017), can effectively improve the performance under label noise.

4.8 Conclusion and Discussion

In this work, we propose NAL for learning with noisy labels. Our method leverages an attention branch and a noise attention loss to learn the attention weights for distinguishing the mislabeled samples from clean samples. NAL can effectively diminish the gradient of mislabeled samples, mitigating the effect of noisy labels. We also provide extensive empirical analyses and evaluate its effectiveness across multiple datasets with different types and ratios of label noises.

There are still multiple open problems for future research. Currently, label noise has been extensively studied in image classification task. Other research areas, such as graph learning and federated learning, remain to be explored. On the methodological front, we hope that our work will trigger interest in designing new network architectures that inherently provide robustness to label noise.

4.9 Appendix

4.9.1 Gradient Derivation of $\ell_{\mathbf{NAL}}$

The sample-wise ℓ_{NAL} can be rewrite as:

$$\ell_{\mathrm{NAL}} = \ell_{\mathrm{na}} + \lambda \ell_{\mathrm{b}} = -\sum_{k=1}^{K} q_k \log(\tau(p_k - q_k) + q_k) - \lambda \log \tau. \tag{4.7}$$

As τ is learned from attention branch. The derivation of the ℓ_{NAL} with respect to the logits is as follows:

$$\frac{\partial \ell_{\text{NAL}}}{\partial z_j} = \frac{\partial \ell_{\text{na}}}{\partial z_j} = -\sum_{k=1}^{K} \frac{\tau q_k}{\tau(p_k - q_k) + q_k} \frac{\partial p_k}{\partial z_j}. \tag{4.8}$$

Since $p_k = \texttt{softmax}(\boldsymbol{z}) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$, we have

$$\frac{\partial p_k}{\partial z_j} = \frac{\partial \left(\frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}\right)}{\partial z_j} = \frac{\frac{\partial e^{z_k}}{\partial z_j}\left(\sum_{j=1}^{K} e^{z_j}\right) - e^{z_k} \frac{\partial \left(\sum_{j=1}^{K} e^{z_j}\right)}{\partial z_j}}{\left(\sum_{j=1}^{K} e^{z_j}\right)^2}. \tag{4.9}$$

In the case of $k = j$:

$$\begin{aligned}\frac{\partial p_k}{\partial z_j} =& \frac{\frac{\partial e^{z_k}}{\partial z_k}\left(\sum_{k=1}^{K} e^{z_k}\right) - e^{z_k} \frac{\partial \left(\sum_{k=1}^{K} e^{z_k}\right)}{\partial z_k}}{\left(\sum_{k=1}^{K} e^{z_k}\right)^2} = \frac{e^{z_k}\left(\sum_{k=1}^{K} e^{z_k}\right) - e^{z_k} \cdot e^{z_k}}{\left(\sum_{k=1}^{K} e^{z_k}\right)^2} \\ =& \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}} - \left(\frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}\right)^2 = p_k - p_k^2. \end{aligned} \tag{4.10}$$

In the case of $k \neq j$:

$$\frac{\partial p_k}{\partial z_j} = \frac{0 \cdot \left(\sum_{j=1}^{K} e^{z_j}\right) - e^{z_k} \cdot e^{z_j}}{\left(\sum_{j=1}^{K} e^{z_j}\right)^2} = -\frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}} \frac{e^{z_j}}{\sum_{j=1}^{K} e^{z_j}} = -p_k p_j. \tag{4.11}$$

Combining Eq. (4.10) and (4.11) into Eq. (4.8), we obtain:

$$\begin{aligned}\frac{\partial \ell_{\mathrm{NAL}}}{\partial z_j} =& -\sum_{k=1}^{K} \frac{\tau q_k}{\tau(p_k - q_k) + q_k} \frac{\partial p_k}{\partial z_j} \\ =& -\frac{\tau q_j}{\tau(p_j - q_j) + q_j} \frac{\partial p_j}{\partial z_j} - \sum_{k \neq j}^{K} \frac{\tau q_k}{\tau(p_k - q_k) + q_k} \frac{\partial p_k}{\partial z_j} \\ =& -\frac{\tau q_j}{\tau(p_j - q_j) + q_j}(p_j - p_j^2) - \sum_{k \neq j}^{K} \frac{\tau q_k}{\tau(p_k - q_k) + q_k}(-p_k p_j) \\ =& -\frac{\tau q_j p_j}{\tau(p_j - q_j) + q_j} + p_j \sum_{k=1}^{K} \frac{\tau q_k p_k}{\tau(p_k - q_k) + q_k}. \end{aligned} \tag{4.12}$$

Therefore, if $q_j = q_y = 1$, then

$$\frac{\partial \ell_{\mathrm{NAL}}}{\partial z_j} = -\frac{\tau p_j}{\tau p_j - \tau + 1} + p_j \frac{\tau q_j p_j}{\tau(p_j - 1) + 1} = (p_j - 1)\frac{\tau p_j}{\tau p_j - \tau + 1} = (p_j - 1)\frac{p_j}{p_j - 1 + 1/\tau}. \tag{4.13}$$

If $q_j = 0$, then

$$\frac{\partial \ell_{\mathrm{NAL}}}{\partial z_j} = p_j \frac{\tau q_y p_y}{\tau(p_y - q_y) + q_y} = p_j \frac{p_y}{p_y - 1 + 1/\tau}. \tag{4.14}$$

4.9.2 Formal Proofs for Theorem 4.9.1

Theorem 4.9.1. *Given the noise attention loss $\mathcal{L}_{NAL}$, we rewrite the sample-wise loss $\ell_{NAL} = -\sum_{k=1}^{K} q_k \log(\tau(p_k - q_k) + q_k) - \lambda \log \tau$. Its gradient with respect to the logits z_j can be derived as*

$$\frac{\partial \ell_{NAL}}{\partial z_j} = \begin{cases} \dfrac{p_y}{p_y - 1 + 1/\tau}(p_j - 1) \leq 0, & q_j = q_y = 1 \ (j \textit{ is the true class for } \boldsymbol{x}) \quad (4.15a) \\ \dfrac{p_y}{p_y - 1 + 1/\tau} p_j \geq 0, & q_j = 0 \ (j \textit{ is not the true class for } \boldsymbol{x}) \quad (4.15b) \end{cases}$$

where the multiplier $\frac{p_y}{p_y-1+1/\tau} \in (0,1)$.

Proof. From the Appendix 4.9.1, we have the gradient of the sample-wise ℓ_{NAL} with respect to the logits z_j as

$$\frac{\partial \ell_{\text{NAL}}}{\partial z_j} = -\sum_{k=1}^{K} \frac{\tau q_k}{\tau(p_k - q_k) + q_k} \frac{\partial p_k}{\partial z_j} \tag{4.16}$$

where $\frac{\partial p_k}{\partial z_j}$ can be further derived base on whether $k = j$ by follows:

$$\frac{\partial p_k}{\partial z_j} = \begin{cases} p_k - p_k^2 & k = j \\ -p_j p_k & k \neq j \end{cases} \tag{4.17}$$

According to Eq. (4.16) and (4.17), the gradient of ℓ_{NAL} can be derived as:

$$\frac{\partial \ell_{\text{NAL}}}{\partial z_j} = \begin{cases} \frac{p_j}{p_j-1+1/\tau}(p_j - 1) = \frac{p_y}{p_y-1+1/\tau}(p_j - 1), & q_j = q_y = 1 \\ \\ \frac{p_y}{p_y-1+1/\tau} p_j, & q_j = 0 \end{cases} \tag{4.18}$$

We denote $\varphi = \frac{p_y}{p_y-1+1/\tau}$. Since $p_j \leq 1$, we have $p_j - 1 \leq 0$. As $\tau \in (0,1)$, the term $\frac{p_y}{p_y-1+1/\tau} \in (0,1)$, we have $(p_j - 1)\frac{p_y}{p_y-1+1/\tau} \leq 0$ and $p_j \frac{p_y}{p_y-1+1/\tau} \geq 0$. □

4.9.3 Detail Description of Experiments

Source code for the experiments is available in the zip file. All experiments are implemented in PyTorch and run in a single Nvidia A100 GPU. For CIFAR-10 and CIFAR-100, we do not perform early stopping since we don't assume the presence of clean validation data. All test accuracy are recorded from the last epoch of training.

For Clothing1M, it provides 50k, 14k, 10k refined clean data for training, validation and testing respectively. Note that we do not use the 50k clean data for fair comparison with existing methods. Similar to the compared methods (Liu *et al.*, 2020; Wang *et al.*, 2019), we report the test accuracy when the performance on validation set is optimal. All tables of CIFAR-10/CIFAR-100 report the mean and standard deviation from 3 trails with different random seeds for different simulated noise.

Dataset Description and Preprocessing

The information of datasets are described in Table 1.1. CIFAR-10 and CIFAR-100 are clean datasets, we describe the label noise injection in Appendix 4.9.3. ANIMAL-10N contains human-labeled online images for 10 animals with confusing appearance. Its estimated noise rate is 8%. Clothing1M consists of 1 million training images from 14 categories collected from online shopping websites with noisy labels generated from surrounding texts. Its noise level is estimated as 38.5%. Following (Jiang *et al.*, 2018; Chen *et al.*, 2019), we use the mini WebVision dataset which contains the top 50 classes from the Google image subset of WebVision, which results in approximate 66 thousand images. The noise level of WebVision is estimated at 20% (Li *et al.*, 2017).

As for data preprocessing, we apply normalization and regular data augmentation (i.e. random crop and horizontal flip) on the training sets of all datasets. The cropping size is consistent with existing works (Liu *et al.*, 2020; Li *et al.*, 2020a). Specifically, 32 for CIFAR-10 and CIFAR-100, 224 $\times$ 224 for Clothing 1M (after resizing to 256 $\times$ 256), and 227 $\times$ 227 for Webvision.

Simulated Label Noise Injection

Since the CIFAR-10 and CIFAR-100 are initially clean, we follow (Tanaka *et al.*, 2018; Patrini *et al.*, 2017) for symmetric and asymmetric label noise injection. Specifically, symmetric label noise is generated by randomly flipping a certain fraction of the labels in the training set following a uniform distribution. Asymmetric label noise is simulated by flipping their class to another certain class according to the mislabel confusions in the real world. For CIFAR-10, the asymmetric noisy labels are generated by mapping *truck* $\rightarrow$ *automobile*, *bird* $\rightarrow$ *airplane*, *deer* $\rightarrow$ *horse* and *cat* $\leftrightarrow$ *dog*. For CIFAR-100, the noise flips each class into the next, circularly within super-classes.

Training Procedure

CIFAR-10/CIFAR-100: We use a ResNet-34 and train it using SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 64. The network is trained for 500 epochs for both CIFAR-10 and CIFAR-100. We use the cosine annealing learning rate (Loshchilov and Hutter, 2016) where the maximum number of epoch for each period is 10, the maximum and minimum learning rate is set to 0.02 and 0.001 respectively. Note that the reason that we train the model 500 epochs in total is not because of the slow convergence (Our method actually converges around 250 epochs, shown in Section 4.9.6). Instead, it is to fully evaluate whether the proposed method will overfit mislabeled samples, which avoids the interference caused by early stopping Li *et al.* (2020b) (i.e. the model may not start overfitting mislabeled samples when the number of training epochs is small, especially when learning rate scheduler is cosine annealing (Loshchilov and Hutter, 2016)).

Clothing1M: Following (Xiao *et al.*, 2015; Wang *et al.*, 2019), we use a ResNet-50 without pretrained parameters. We train the model with batch size 64. The optimization is done using SGD with a momentum 0.9, and weight decay 0.001. We use the same cosine annealing learning rate as CIFAR-10 except the minimum learning rate is set to 0.0001 and total epoch is 250. For each epoch, we sample 2000 mini-batches from the training data ensuring that the classes of the noisy labels are balanced.

Webvision: Following (Li *et al.*, 2020a; Liu *et al.*, 2020), we use an Inception-ResNetV2 as the backbone architecture. All other optimization details are the same as for CIFAR-10, except for the weight decay (0.0005) and the batch size (32).

Hyperparameters Selection and Sensitivity

We perform hyperparameter tuning via grid search: $\lambda = [0.01, 0.05, 0.1, 0.5, 5, 10, 50]$ and $\alpha = [0.7, 0.9, 0.99]$ using a noisy validation set sampled from the noisy training set, which is similar to (Liu *et al.*, 2020). In our experiments, we set $\alpha = 0.9$ for all datasets. For CIFAR-10, we set $\lambda = 0.5$. For CIFAR-100, we set $\lambda = 10$. For ANIMAL-10N, we set $\lambda = 0.5$. For Webvision and Clothing1M, we set $\lambda = 50$.

4.9.4 More Results on Training Clothing1M from Scratch

Existing methods use the ResNet-50 pretrained on ImageNet. Here we report the results when existing methods train the model from scratch in Table 3.5. We run the official code of SCE, SELC, DivideMix and ELR+. SCE (Wang *et al.*, 2019) is a noise robust loss function. SELC (Lu and He, 2022) is a label correction method. These two methods only focus on modifying the loss function, thus no extra GPU space

is required. In contrast, DivideMix and ELR+ (i.e. an improved version of ELR) are complex method that use multiple techniques to boost their performance. Both of them use the two networks and mixup augmentations. Other technique, such as weight average is also applied in ELR+. Therefore, more GPU space is required for DivideMix and ELR+. As we can observe in Table 4.6, the proposed method NAL still outperforms these methods when training the ResNet-50 from scratch.

Method	Batch Size	Required GPU Memory	Accuracy
SCE† (Wang *et al.*, 2019)	64	7.39 GB	71.02
SCE‡ (Wang *et al.*, 2019)	64	7.39 GB	69.40
SELC† (Lu and He, 2022)	64	7.39 GB	74.01
SELC‡ (Lu and He, 2022)	64	7.39 GB	72.02
ELR† (Liu *et al.*, 2020)	64	-	72.87
ELR+† (Liu *et al.*, 2020)	64	21.70 GB	74.81
ELR+‡ (Liu *et al.*, 2020)	64	21.70 GB	72.80
DivideMix† (Li *et al.*, 2020a)	32	19.03 GB	74.76
DivideMix‡ (Li *et al.*, 2020a)	32	19.03 GB	70.29
NAL‡ (ours)	64	7.46 GB	73.58

Table 4.6: The accuracy (%) results on Clothing1M. The marker † denotes the methods use ResNet-50 pretrained on ImageNet. The marker ‡ denotes the model is trained from scratch.

4.9.5 More results of Estimated Targets

We report the confusion matrix for other levels of label noise (symmetric 40%, 80% and asymmetric 40%) on CIFAR-10. Figure 4.5, Figure 4.7 and Figure 4.9 display the confusion matrix of noisy labels w.r.t. the clean labels on CIFAR-10 with 40% symmetric, 80% symmetric and 40% asymmetric label noise respectively. Figure 4.6, Figure 4.8 and Figure 4.10 display the confusion matrix of corrected labels w.r.t. the clean labels on CIFAR-10 with 40% symmetric, 80% symmetric and 40% asymmetric

label noise after using the proposed method, respectively. As we can observe, the corrected labels (estimated targets) yield better quality than the original noisy labels, even under the extreme label noise (e.g. 80% symmetric noise).

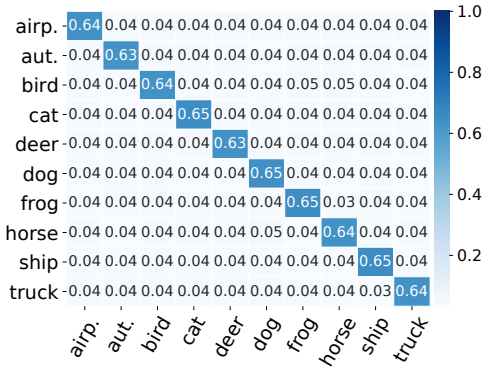


Figure 4.5: Confusion matrix of noisy labels w.r.t clean labels on CIFAR-10 with 40% symmetric label noise.

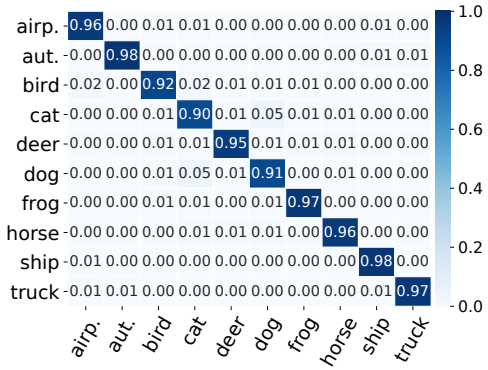


Figure 4.6: Confusion matrix of corrected labels w.r.t clean labels on CIFAR-10 with 40% symmetric label noise.

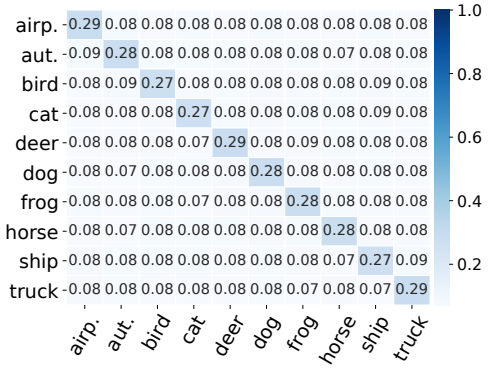


Figure 4.7: Confusion matrix of noisy labels w.r.t clean labels on CIFAR-10 with 80% symmetric label noise.

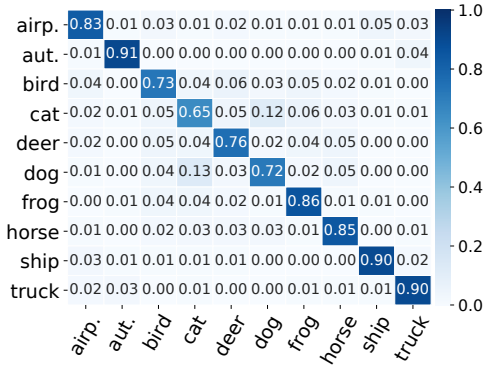


Figure 4.8: Confusion matrix of corrected labels w.r.t clean labels on CIFAR-10 with 80% symmetric label noise.

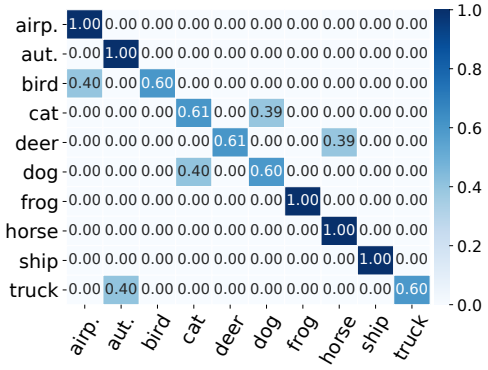


Figure 4.9: Confusion matrix of noisy labels w.r.t clean labels on CIFAR-10 with 40% asymmetric label noise.

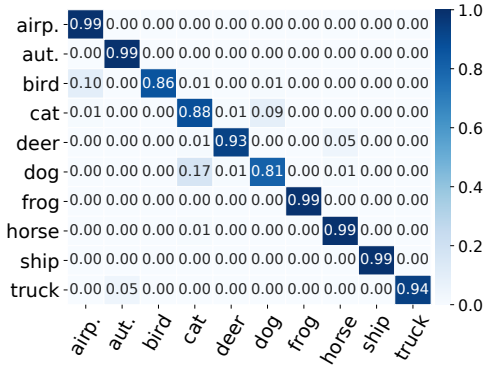


Figure 4.10: Confusion matrix of corrected labels w.r.t clean labels on CIFAR-10 with 40% asymmetric label noise.

4.9.6 Accuracy Curves versus Epochs

In this section, we show the accuracy curves during training to show the noise robustness of NAL. As we can observe in Figure 4.11, CE fits the whole noisy labels eventually, while NAL only fits around 60% training samples. In Figure 4.12, the test accuracy of CE decreases due to memorization of noisy labels, while NAL won't. We also observe the similar results on CIFAR-10.

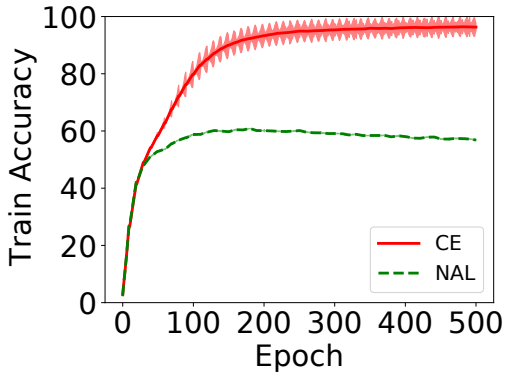


Figure 4.11: Train accuracy vs. epochs on CIFAR-100 with 40% symmetric label noise.

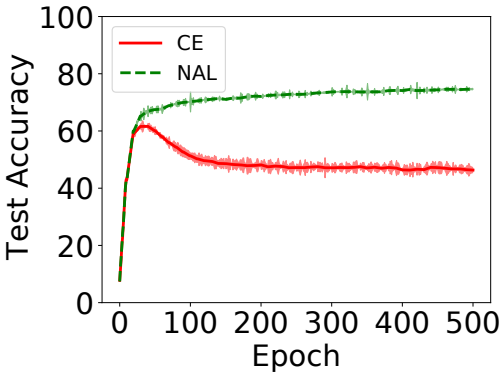


Figure 4.12: Test accuracy vs. epochs on CIFAR-100 with 40% symmetric label noise.

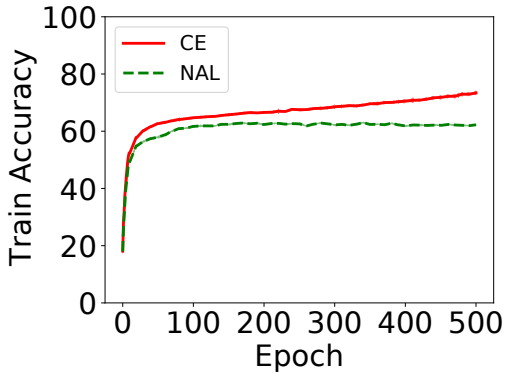


Figure 4.13: Train accuracy vs. epochs on CIFAR-10 with 40% symmetric label noise.

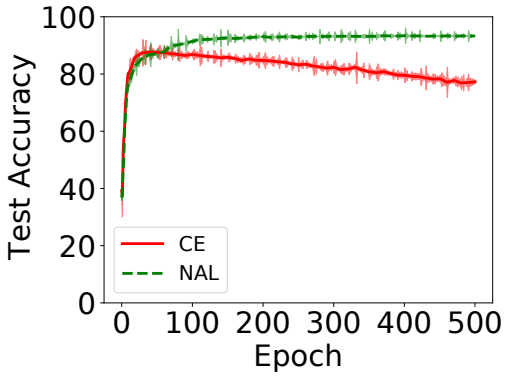


Figure 4.14: Test accuracy vs. epochs on CIFAR-10 with 40% symmetric label noise.

Chapter 5

Entropy Minimization for Learning with Noisy Labels

5.1 Citation and Main Contributor

Lu, Yangdi and He, Wenbo. "Entropy Minimization for Deep Learning with Noisy Labels." Submitted to AAAI 2023.

The main contributor to this paper is the first author - Yangdi Lu (contributes more than 80%).

5.2 Introduction

Deep neural networks (DNNs) have demonstrated phenomenal success in multimedia research and are widely adopted in image classification applications, such as recognition of retail products and automatic monitoring of biodiversity. In traditional image classification tasks, we always expect and assume a perfectly labeled training set.

However, in practice, obtaining a large-scale dataset with fully correct (clean) labels is usually expensive, and sometimes even impossible. The widely-used approaches for constructing datasets involve some degree of automatic labeling or crowdsourcing, which are inherently error-prone (Sambasivan *et al.*, 2021). Unsupervised and semi-supervised learning techniques aim to alleviate this requirement by incorporating unlabeled examples, but cannot be directly applied to learning with *noisy labels*. Previous studies (Zhang *et al.*, 2018a; Arpit *et al.*, 2017) have demonstrated that noisy training data degenerate the robustness of learned models, especially for DNNs which are prone to overfitting label noise, leading to poor generalization performance. A recent study (Northcutt *et al.*, 2021) reports that pervasive label errors even occur in the test sets of most commonly-used benchmarks. Therefore, studying noisy labels and developing techniques for training accurate DNNs in the presence of noisy labels is of great practical significance.

The dominant approaches to learning with noisy labels in recent years include: 1) loss correction methods (Sukhbaatar *et al.*, 2015; Goldberger and Ben-Reuven, 2017; Patrini *et al.*, 2017) seek to correct the loss by estimating the noise transition matrix; 2) label correction (Tanaka *et al.*, 2018; Arazo *et al.*, 2019; Li *et al.*, 2020a; Kim *et al.*, 2019) or label filtering methods (Thulasidasan *et al.*, 2019a; Nguyen *et al.*, 2020; Pleiss *et al.*, 2020) aim to identify and correct the mislabeled samples or remove the mislabeled samples; 3) sample selection methods (Han *et al.*, 2018; Yu *et al.*, 2019; Lu *et al.*, 2022) focus on training DNNs only using selected "clean" samples; 4) robust loss functions (Ghosh *et al.*, 2017; Zhang and Sabuncu, 2018; Wang *et al.*, 2019; Ma *et al.*, 2020) are specifically designed to achieve noise-robustness in the presence of noisy labels. Different from the first three categories which often lead to

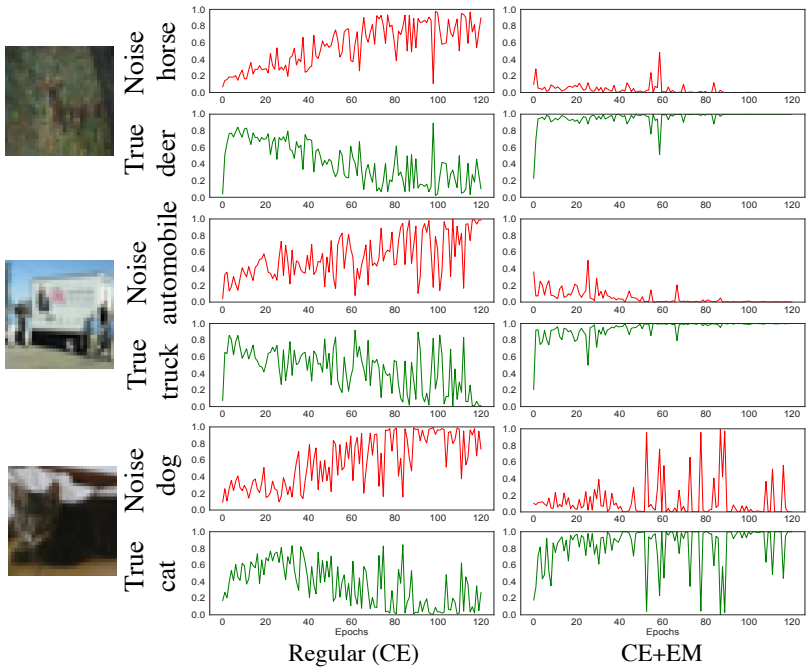


Figure 5.1: Softmax outputs on noisy label and latent true label when training an 8-layer CNN on CIFAR10 with 40% asymmetric label noise. The x-axis is the training epochs and the y-axis is the output probability on assigned label. We compare the training using cross entropy (CE) loss with the proposed method that adds entropy minimization (EM) to CE. The output probability of CE+EM is more stable than CE.

complicated training procedures, such as maintaining multiple models or alternating between updating the model and updating the training set, robust loss functions provide the simplest yet universal solution.

When training DNNs with a noisy training set consisting of clean and mislabeled samples, it has been observed that the model outputs tend to fluctuate, especially for mislabeled samples. For a better explanation, we conduct an empirical study in which the model is trained with CIFAR-10 under 40% asymmetric label noise (e.g. 40% deer/truck/cat images are mislabeled as horse/automobile/dog respectively). We

then plot the entry of softmax output corresponding to the noisy label and true label throughout the training in Figure 5.1. During the regular training using cross entropy (CE) loss, the outputs can vibrate with large oscillations. Take the first row as an example, a deer image is mislabeled as a horse. In the training, the model begins with a high probability to indicate it is a deer image since the clean deer samples would encourage the model to predict this deer image as a deer. However, with the training continuing, the deer samples with horse labels pull the model back to predict this deer image as a horse, thus the model has a larger and larger probability to indicate it is a horse image. As a consequence, the model eventually memorizes the wrong labels. In addition, recent works (Liu, 2021; Cheng *et al.*, 2020) have observed that with the presence of label noise, learning with noisy labels directly eventually results in unconfident model predictions. Wei *et al.* (2021b) also demonstrated that improving the confidence of model prediction leads to better performance.

Inspired by these observations, we focus on stabilizing the model outputs to make predictions with high confidence and less fluctuation, resulting in combating noisy labels. Briefly, we use the concept of *entropy* from information theory to adjust the confidence of predictions. Different from the previous studies (Pereyra *et al.*, 2017; Li *et al.*, 2020a) that aim to maximize the entropy for penalizing confident predictions, we propose to minimize the entropy of predictions. Specifically, we simply add a weighted entropy term to CE loss during training. Intuitively, minimizing entropy constricts the randomness of the model predictions, allowing the model to produce consistent predictions during the training. The right column in Figure 5.1 shows the effect after adding the entropy term to CE. We observe that the outputs on the latent true label become more stable compared to using CE, even for the most

ambiguous scenario (i.e. the third case cat $\rightarrow$ dog). To the best of our knowledge, entropy minimization was first proposed for semi-supervised learning (Grandvalet and Bengio, 2005) and recently was also applied for domain adaptation (Wu *et al.*, 2021), while our work is the first to apply it for learning with noisy labels.

From the theoretical point, we establish the connection between our approach and the recent findings in robust loss functions. We prove that a non-robust loss is tolerant to label noise when applying entropy minimization. In addition, the proposed approach would not suffer from the underfitting problem as it inherits the fitting ability from non-robust loss functions. This advantage surpasses most existing noise-robust loss functions, such as Mean Absolute Error (MAE) (Ghosh *et al.*, 2017) and Reverse CE (RCE) (Wang *et al.*, 2019), which have been observed to suffer from the underfitting problem on complicated datasets (Zhang and Sabuncu, 2018; Ma *et al.*, 2020). Furthermore, our empirical results on benchmark datasets and real-world datasets demonstrate that the proposed approach achieves better robustness than the state-of-the-art methods. Our main contributions are summarized as follows:

- We provide new insights into robust learning with noisy labels by reducing the model output fluctuations (or improving the output confidence). Specifically, we propose to minimize the entropy of prediction to enhance learning with label noise.
- We theoretically prove that by applying entropy minimization to non-robust loss functions, they can be robust to noisy labels without suffering from the underfitting problem under certain noise rate.
- We empirically demonstrate that the proposed loss functions outperform the state-of-the-art robust loss functions by a substantial margin.

5.3 Methodology

In this section, we first introduce the preliminaries in learning with noisy labels. Subsequently, we review the noise tolerance of existing loss functions according to symmetric condition (Ghosh *et al.*, 2017) and explain their underfitting problem by gradient analysis. Then, to achieve robustness and avoid the underfitting problem, we propose a simple and effective approach that can be applied to any loss function.

5.3.1 Preliminaries

Supervised Classification. We consider the problem of K-class classification. Given a dataset $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^d$ is a d-dimensional feature input and $y \in \mathcal{Y} = [K] = \{1, \dots, K\}$ is the annotated label. A DNN classifier f ending with a softmax layer is a function that maps feature space to label space: $f : \mathcal{X} \to \Delta^{K-1}$, where Δ^{K-1} denotes the K-dimensional simplex. We denote the output probability distribution $f(\boldsymbol{x}) = [p(1|\boldsymbol{x}), ..., p(K|\boldsymbol{x})], p(k|\boldsymbol{x}) = \frac{e^{(\boldsymbol{z})_k}}{\sum_{j=1}^K e^{(\boldsymbol{z})_j}}, k \in \{1, \dots, K\}$, where $(\boldsymbol{z})_k$ denotes the logits with respect to class k. Given the ground-truth distribution over different classes for input $\boldsymbol{x}$ is $q(k|\boldsymbol{x})$, $\sum_{k=1}^K q(k|\boldsymbol{x}) = 1$ and y is the true label, we have $q(y|\boldsymbol{x}) = 1$ and $q(k|\boldsymbol{x}) = 0, \forall k \neq y$. In this way, the commonly used CE loss can be represented as:

$$\ell_{\text{ce}}(f(\boldsymbol{x}), y) = -\sum_{k=1}^{K} q(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}) = -\log p(y|\boldsymbol{x}) \tag{5.1}$$

Label Noise Models. The generation of real-world label noise is unpredictable, a popular methodology to cope with noisy labels is to hypothesize noise models then

develop robust algorithms based on them. Two popular noise assumptions are *symmetric* and *asymmetric* label noise (Natarajan *et al.*, 2013), where the noise is corrupted by a $K \times K$ noise transition matrix Q and independent to the input feature, i.e., $Q_{ij} = P(\hat{y} = j|y = i) = P(\hat{y} = j|\boldsymbol{x}, y = i)$, where y denotes the true label and $\hat{y}$ denotes the noisy label. Suppose the overall noise rate as η, for symmetric noise, the flip probability to other labels is constant, i.e., $Q_{ij} = 1 - \eta$ for $i = j$ and $Q_{ij} = \frac{\eta}{K-1}$ for $i \neq j$. For asymmetric noise, it is a simulation of real-world label noise, where labels are only replaced by similar classes (e.g. dog↔cat).

Learning with Noisy Labels and Noise Tolerance. Our goal is to learn from a noisy training set $\hat{D} = \{(\boldsymbol{x}_i, \hat{y}_i)\}_{i=1}^n$ where the labels are corrupted. Given any loss function $\ell(f(\boldsymbol{x}), y)$, we define the risk of f under clean distribution as $R_\ell(f) = \mathbb{E}_D[\ell(f(x), y)]$, and under noisy distribution as $R_\ell^\eta(f) = \mathbb{E}_{\hat{D}}[\ell(f(x), \hat{y})]$. Let f^* and f_η^* be the global minimizers of $R_\ell(f)$ and $R_\ell^\eta(f)$ respectively. Assume the label noise is symmetric and the noise rate is η, then we have corresponding risk under label noise as

$$R_\ell^\eta(f) = \mathbb{E}_D[(1-\eta)\ell(f(\boldsymbol{x}), y) + \frac{\eta}{K-1}\sum_{i \neq y}\ell(f(\boldsymbol{x}), i)] \tag{5.2}$$

We say that the risk minimization under a loss function is noise-tolerant if $R_\ell^\eta(f)$ shares the same global minimum as $R_\ell(f)$.

5.3.2 Existing Robust Loss Functions

Suppose symmetric label noise rate $\eta < \frac{K-1}{K}$, a loss function is proved to be noise-tolerant for K-class classification task if the loss function satisfies the symmetric

Name	Loss expression	$\sum_{i=1}^{K} \ell(f(\boldsymbol{x}), i)$	Symmetric	Gradient $\frac{\partial \ell(f(x),y)}{\partial \theta}$
CE	$-\log p(y\|\boldsymbol{x})$	$-\sum_{i=1}^{K} \log p(i\|\boldsymbol{x})$	×	$-\frac{1}{p(y\|\boldsymbol{x})}\nabla_\theta p(y\|\boldsymbol{x})$
FL	$-(1-p(y\|\boldsymbol{x}))^\gamma \log p(y\|\boldsymbol{x})$	$-\sum_{i=1}^{K}(1-p(i\|\boldsymbol{x}))^\gamma \log p(i\|\boldsymbol{x})$	×	$\Big[\gamma(1-p(y\|\boldsymbol{x}))^{\gamma-1}\log p(y\|\boldsymbol{x}) - \frac{(1-p(y\|\boldsymbol{x}))^\gamma}{p(y\|\boldsymbol{x})}\Big]\nabla_\theta p(y\|\boldsymbol{x})$
MAE	$2(1-p(y\|\boldsymbol{x}))$	$2K-2$	√	$-2\nabla_\theta p(y\|\boldsymbol{x})$
RCE	$-A(1-p(y\|\boldsymbol{x}))$	$-AK+A$	√	$A\nabla_\theta p(y\|\boldsymbol{x})$
GCE	$\frac{1-p(y\|\boldsymbol{x})^\rho}{\rho}$	$\big[\frac{K-K^{1-\rho}}{\rho}, \frac{K-1}{\rho}\big]$	√	$-\frac{1}{p(y\|\boldsymbol{x})^{1-\rho}}\nabla_\theta p(y\|\boldsymbol{x})$
TCE	$\sum_{i=1}^{t} \frac{(1-p(y\|\boldsymbol{x}))^i}{i}$	$\big[K-1, (K-1)\sum_{i=1}^{t}\frac{1}{i}\big]$	√	$-\frac{1-(1-p(y\|\boldsymbol{x}))^t}{p(y\|\boldsymbol{x})}\nabla_\theta p(y\|\boldsymbol{x})$
NCE	$\log_{\Pi_k^K p(k\|\boldsymbol{x})} p(y\|\boldsymbol{x})$	1	√	$\frac{\sum_{k\neq y}^{K}\log p(k\|\boldsymbol{x})}{(\sum_{k=1}^{K}\log p(k\|\boldsymbol{x}))^2}\cdot\frac{1}{p(y\|\boldsymbol{x})}\nabla_\theta p(y\|\boldsymbol{x})$

Table 5.1: Comparison of existing loss functions in learning with noisy labels. For FL, parameter $\gamma \geq 0$ and FL reduces to the CE loss when $\gamma = 0$. For RCE, A is a negative constant to replace $\log(0)$. For GCE, parameter $\rho \in (0, 1]$. For TCE, parameter $t \in \mathbb{N}_+$. For deriving gradients, θ denotes the parameters of f.

condition as follows (Ghosh *et al.*, 2017):

$$\sum_{k=1}^{K} \ell(f(\boldsymbol{x}), k) = C, \forall \boldsymbol{x} \in \mathcal{X}, \forall f \in \mathcal{H}, \tag{5.3}$$

where C is a constant, and $\mathcal{H}$ is the hypothesis class. Combined Eq. (5.2) and Eq. (5.3), we can easily derive

$$R_\ell^\eta(f) = \big(1 - \frac{\eta K}{K-1}\big) R_\ell(f) + \frac{\eta C}{K-1} \tag{5.4}$$

Since $1 - \frac{\eta K}{K-1} > 0$, if f^* is the global minimizer of $R_\ell(f)$, then it is also the minimizer of $R_\ell^\eta(f)$. Therefore, a symmetric loss function is theoretical noise-tolerant if the global minimizer can be learned. However, the derivation of global optimum is a strong assumption. In practice, many symmetric loss functions, such as MAE and RCE, have been observed to suffer from the underfitting problem on complicated datasets (Zhang and Sabuncu, 2018; Ma *et al.*, 2020). We review the existing loss functions and derive their gradients in Table 5.1. The CE loss and focal loss (FL) (Lin *et al.*, 2017) are not robust to noisy labels but have the advantage of sufficient learning ability.

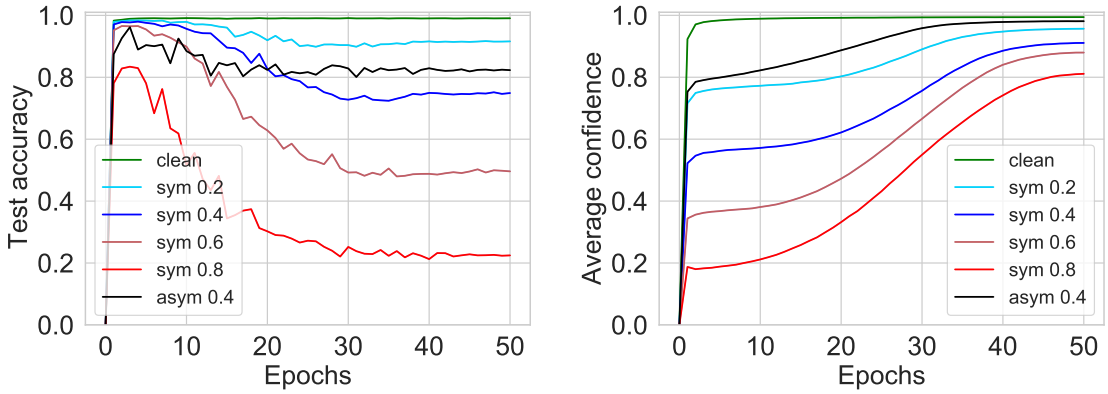
Both of them put more weights on the gradient of ambiguous (hard) samples. On the contrary, MAE and Reverse CE (RCE) (Wang *et al.*, 2019) are robust to noisy labels but increase difficulty in training as they equally provide the same weights on the gradient for all training samples. To balance learning sufficiency and noise robustness, a generalized version of CE loss (GCE) (Zhang and Sabuncu, 2018) was proposed $\ell_{\text{gce}} = \frac{1-p(y|\boldsymbol{x})^{\rho}}{\rho}$, which reduces to MAE and CE when $\rho = 1$ and $\rho \to 0$, respectively. Similarly, Taylor cross entropy (TCE) (Feng *et al.*, 2021) loss was proposed $\ell_{\text{tce}} = \sum_{i=1}^{t} \frac{(1-p(y|\boldsymbol{x}))^i}{i}$, which is also a generalized mixture of CE (when $t \to \infty$) and MAE (when $t = 1$). Other loss function such as symmetric cross entropy (SCE) (Wang *et al.*, 2019) that combines CE with RCE to achieve partial robust to label noise without sacrificing learning ability. Recently, Ma *et al.* have demonstrated that any loss can be made robust to noisy labels by applying a simple normalization and proposed the normalized cross entropy (NCE) (Ma *et al.*, 2020). However, the normalization operation actually alters the gradient of CE loss so that NCE no longer retains the original fitting ability. Let's denote $P = \log p(y|\boldsymbol{x})$ and $Q = \sum_{k \neq y} \log p(k|\boldsymbol{x})$. In Table 5.1, the gradient of NCE is weighted by the term $\frac{\sum_{k\neq y}^{K} \log p(k|\boldsymbol{x})}{(\sum_{k=1}^{K} \log p(k|\boldsymbol{x}))^2} = \frac{Q}{(P+Q)^2}$. During training, the Q term may increase even when P is fixed. Q reaches the maximum value when all $p(k \neq y|\boldsymbol{x})$ equals to $(1 - p(y|\boldsymbol{x}))/(K-1)$. As a consequence, the corresponding gradient reaches the minimum value, which hinders the convergence and causes the underfitting problem. To remedy this problem, Active Passive Loss (APL) (Ma *et al.*, 2020) was proposed for both robust and sufficient learning by combining two loss terms, i.e., a robust active loss term and a robust passive loss term.

5.3.3 Robust Learning by Entropy Minimization

Existing methods focus on creating new loss functions that meet the symmetric condition while simultaneously solving the underfitting problem by combining multiple robust loss terms. Different from them, we study the entropy of model output for noise robustness.

In classification with clean training data, the confidence score of a prediction given $\boldsymbol{x}$ is calculated by $\max_j p(j|\boldsymbol{x})$. Namely, the probability that the assigned label is correct. A classifier is overconfident when it places all probability on a single class, which is often a symptom of overfitting (Szegedy *et al.*, 2016). Existing work (Pereyra *et al.*, 2017) has proposed to regularize the classifier by penalizing high-confident output distributions, leading to better generalization. In classification with noisy labels, we observe that overfitting to label noise does not attribute to overconfident predictions. On the contrary, the average confidence of predictions decreases due to fluctuation of predictions when training with noisy labels. Figure 5.2 shows the test accuracy and corresponding average confidence on MINIST with different label noises. As can be observed, in addition to the drop in test accuracy, the average confidence of model predictions in training data also becomes smaller with the increase of label noise rate.

Inspired by this observation, we add an entropy term to the non-robust loss functions to restrict the model to produce high-confidence predictions. Given any input $\boldsymbol{x}$, the model outputs a probability distribution $f(\boldsymbol{x}) = [p(1|\boldsymbol{x}), \ldots, p(K|\boldsymbol{x})]$ over different classes through a softmax function. According to information theory, the



(a) Test Accuracy (b) Average Confidence

Figure 5.2: Test accuracy and average confidence on MNIST with various levels of label noise using CE. Learning with label noise not only reduces the test accuracy but also produces predictions with low confidence.

shannon entropy of this probability distribution is calculated by

$$H(f(\boldsymbol{x})) = -\sum_{k=1}^{K} p(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}). \tag{5.5}$$

Confident predictions correspond to output distributions that have low entropy. To force the model produce confident and consistent predictions during the training, we penalize the high entropy prediction, i.e., minimize the entropy of output distribution. Therefore, the loss functions enhanced by entropy minimization (EM) becomes

$$\ell_{\text{EM-enhanced}}(f(\boldsymbol{x}), \hat{y}) = \ell(f(\boldsymbol{x}), \hat{y}) + \lambda H(f(\boldsymbol{x})), \tag{5.6}$$

where $\lambda > 0$ is the hyperparameter. For example, the EM enhanced CE loss becomes

$$\ell_{\text{CE+EM}}(f(\boldsymbol{x}), \hat{y}) = -\log p(\hat{y}|\boldsymbol{x}) - \lambda \sum_{k=1}^{K} p(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}), \tag{5.7}$$

5.4 Theoretical Justification

In this section, we first demonstrate the noise robustness of the EM-enhanced loss functions. Then we provide the analysis of how our method avoids the underfitting problem.

5.4.1 Noise Robustness

The minimization of the entropy term (Eq. (5.5)) could push the network prediction $f(\boldsymbol{x})$ towards any vertex v of Δ^{K-1}, namely, $f(\boldsymbol{x}) = v = [0, \ldots, 1, \ldots, 0]$, which results the minimum value of entropy (zero). To clarify this scenario, we assume:

Assumption 5.4.1. *In a multi-class classification problem, the original model output* $f(\boldsymbol{x}) \in \Delta^{K-1}$. *When optimizing* $\min_{\theta} \big[\ell(f(\boldsymbol{x}), \hat{y}) + \lambda H(f(\boldsymbol{x}))\big]$, *the resulted hypothesis* $f \in \mathcal{H}_v : \mathcal{X} \to V$ *such that the output* $f(x) \in V$, *where* V *denotes the vertices set of* Δ^{K-1}.

Based on the Assumption 5.4.1, we have following Lemma to show that performing entropy minimization make non-robust loss functions (e.g. CE and FL) satisfy the symmetric condition.

Lemma 5.4.1. *According to Assumption 5.4.1, the learned hypothesis outputs* $f(x) \in V$. *Given any specific input* $\boldsymbol{x}$, $f(\boldsymbol{x})$ *equals to a specific* $v^* \in V$ *and assume* $0 \log 0 = 0$,

we have

$$\sum_{i=1}^{K} \ell(f(\boldsymbol{x}), i) = \sum_{i=1}^{K} \ell(v^*, i) = C, \tag{5.8}$$

satisfies for any loss ℓ, where C denotes a constant.

Proof sketch. For loss functions without using logarithms, it is easy to show that one-hot prediction leads to a symmetric function. The tricky part is using logarithms. Take CE loss as an example, we have two scenarios: (1) if $\arg\max v^* = \arg\max_j q(j|\boldsymbol{x})$, we have $\sum_{i=1}^{K} \ell_{\text{ce}}(v^*, i) = -(K-1)\log 0$. Similar to clipping operation in (Wang *et al.*, 2019), we denote a negative constant $A = \log 0$. Then we have $C = -A(K-1)$. (2) if $\arg\max v^* \neq \arg\max_j q(j|\boldsymbol{x})$, we have $\sum_{i=1}^{K} \ell_{\text{ce}}(v^*, i) = -K\log 0 = -AK$. Therefore, both scenarios satisfy the symmetric condition.

Lemma 5.4.1 implies that, when performing entropy minimization, the network output $f(\boldsymbol{x})$ is restricted to a vertex v^*, then any loss functions can satisfy the symmetric condition. As mentioned in Section 5.3.2, a symmetric loss function is noise tolerant to both symmetric and asymmetric label noise. Here we have the following theorems to show the robustness of the proposed method under symmetric and asymmetric noise.

Theorem 5.4.1. *In a multi-class classification problem, ℓ is noise tolerant under symmetric label noise if noise rate $\eta < \frac{K-1}{K}$ and $f : \mathcal{X} \to V$. And, given $R_\ell(f^*) = 0$ and $0 \leq \ell(f^*(\boldsymbol{x}), k) \leq \frac{C}{K-1}$, ℓ is also noise tolerant under asymmetric label noise or class-dependent label noise when noise rate $\eta_{yk} < 1 - \eta_y$ with $\sum_{k \neq y} \eta_{yk} = \eta_y$.*

Proof sketch. For symmetric noise, based on and Eq. (5.2) and Eq. (5.8), for any

$f \in \mathcal{H}_v$, we can derive

$$R_\ell^\eta(f) = \big(1 - \frac{\eta K}{K-1}\big) R_\ell(f) + \frac{\eta C}{K-1}. \tag{5.9}$$

Thus, we have

$$R_\ell^\eta(f^*) - R_\ell^\eta(f_\eta^*) = \big(1 - \frac{\eta K}{K-1}\big)(R_\ell(f^*) - R_\ell(f_\eta^*)) \leq 0, \tag{5.10}$$

where f^* and f_η^* is a minimizer of R_ℓ and R_ℓ^η. Since $1 - \frac{\eta K}{K-1} > 0$. and $R_\ell(f^*) - R_\ell(f_\eta^*) \leq 0$, we have $R_\ell^\eta(f^*) - R_\ell^\eta(f_\eta^*) \leq 0$, which proves f^* is also a minimizer of R_ℓ^η. For asymmetric noise, we have

$$R_\ell^\eta(f) = C\mathbb{E}_D(1 - \eta_y) - \mathbb{E}_D\Big[\sum_{k \neq y}(1 - \eta_y - \eta_{yk})\ell(f(\boldsymbol{x}), k)\Big]. \tag{5.11}$$

Let f_η^* and f^* be the minimizer of R_ℓ^η and R_ℓ when $f : \mathcal{X} \to V$, respectively. We have $R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) \leq 0$ and hence derive that

$$\mathbb{E}_D\Big[\sum_{k \neq y}(1 - \eta_y - \eta_{yk})(\ell(f^*(\boldsymbol{x}), k) - \ell(f_\eta^*(\boldsymbol{x}), k))\Big] \leq 0 \tag{5.12}$$

First $(1 - \eta_y - \eta_{yk}) > 0$ as per the assumption $\eta_{yk} < 1 - \eta_y$, we have $\ell(f^*(\boldsymbol{x}), k) \leq \ell(f_\eta^*(\boldsymbol{x}), k)$. Second, our assumption has $R_\ell(f^*) = 0$, we have $\ell(f^*(\boldsymbol{x}), y) = 0$. Since the condition $0 \leq \ell(f^*(\boldsymbol{x}), k) \leq \frac{C}{K-1}$, we have $\ell(f^*(\boldsymbol{x}), k) = \frac{C}{K-1}$ for all $k \neq y$. Then we have $\ell(f_\eta^*(\boldsymbol{x}), k) \geq \frac{C}{K-1}$. For both conditions hold, we have $\ell(f_\eta^*(\boldsymbol{x}), k) = \frac{C}{K-1}$, which implies $\ell(f_\eta^*(\boldsymbol{x}), y) = 0$. Therefore, $\ell(f^*(\boldsymbol{x}), k) = \ell(f_\eta^*(\boldsymbol{x}), k)$ for all $k \in [K]$, thus $f_\eta^* = f^*$ which completes the proof.

Theorem 5.4.1 ensures that the derived hypotheses f_η^* and f^* under noisy labels

and clean labels are the same. Therefore, using entropy minimization does offer an alternative way to achieve noise robustness for non-robust loss functions. However, Assumption 5.4.1 is too strong. It may cause the optimization to fail, especially when using a gradient-based strategy that produces many zero gradients. Therefore, we relax the the output restriction of the hypothesis class with an error bound ϵ in Assumption 5.4.2.

Assumption 5.4.2. *When optimizing* $\min_{\theta} \big[\ell(f(\boldsymbol{x}), \hat{y}) + \lambda H(f(\boldsymbol{x}))\big]$*, the resulted hypothesis* $f \in \mathcal{H}_o : \mathcal{X} \to O$ *such that the output* $|f(x) - v| \leq \epsilon$*, where* $v \in V$*.*

Theorem 5.4.2. *In a multi-class classification problem, let* $f : \mathcal{X} \to O$*. Suppose the loss* ℓ *satisfies* $|\sum_{k=1}^{K}(\ell(f(\boldsymbol{x})_1, k) - \ell(f(\boldsymbol{x})_2, k))| \leq \delta$ *when* $|f(\boldsymbol{x})_1 - f(\boldsymbol{x})_2| \leq \epsilon$*, and* $\delta \to 0$ *when* $\epsilon \to 0$*. Then for symmetric label noise satisfying* $\eta < \frac{K-1}{K}$*, the risk bound can be expressed as*

$$R_\ell(f_\eta^*) - R_\ell(f^*) \leq \frac{2\eta\delta}{K - 1 - \eta K} \tag{5.13}$$

where f_η^* *and* f^* *denote the minimizer of* $R_\ell^\eta(f)$ *and* $R_\ell(f)$ *when* $f \in \mathcal{H}_o$*, respectively.*

Proof sketch. For symmetric label noise, we have

$$R_\ell^\eta(f^*) = (1 - \frac{\eta K}{K-1})R_\ell(f^*) + \frac{\eta}{K-1}\sum_{k=1}^{K}\ell(v, k) + \frac{\eta}{K-1}\delta_1, \tag{5.14}$$

where $\delta_1 = \mathbb{E}_D\Big[\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) - \sum_{k=1}^{K}\ell(v, k)\Big]$. Given $f^* \in \mathcal{H}_o$, we have $|f^*(\boldsymbol{x}) - v| \leq \epsilon$ and $|\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) - \sum_{k=1}^{K}\ell(v, k)| \leq \delta$, which implies $\delta_1 \in [-\delta, \delta]$. Similarly,

we can obtain

$$R_\ell^\eta(f_\eta^*) = (1 - \frac{\eta K}{K-1})R_\ell(f_\eta^*) + \frac{\eta}{K-1}\sum_{k=1}^{K}\ell(v,k) + \frac{\eta}{K-1}\delta_2. \tag{5.15}$$

Based on Eq. (5.14) and Eq. (5.15), we have

$$R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) = (1 - \frac{\eta K}{K-1})(R_\ell(f_\eta^*) - R_\ell(f^*)) + \frac{\eta}{K-1}(\delta_2 - \delta_1). \tag{5.16}$$

Given f_η^* and f^* are the minimizers of $R_\ell^\eta(f)$ and $R_\ell(f)$, we have $R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) \leq 0$. As $1 - \frac{\eta K}{K-1} > 0$, we have

$$R_\ell(f_\eta^*) - R_\ell(f^*) \leq \frac{(\delta_2 - \delta_1)\eta}{K - 1 - \eta K} \leq \frac{2\eta\delta}{K - 1 - \eta K} \tag{5.17}$$

Theorem 5.4.2 indicates that when using entropy minimization following Assumption 5.4.2, the difference of the risks caused by the derived hypotheses f_η^* and f^* under noisy labels and clean labels are always bounded. The bound are related to the parameter ϵ and η. When $\epsilon \to 0$ or $\eta \to 0$, the bound tends to 0.

5.4.2 Avoid Underfitting Problem

Noise robust alone is not sufficient. Many existing loss functions (e.g. MAE, RCE and NCE) have been observed to suffer from an underfitting problem. Here we explain how our approach preserves the learning sufficiency by analyzing the gradient. Consider the CE+EM loss $\ell = -\log p(y|\boldsymbol{x}) - \lambda \sum_i^K p(i|\boldsymbol{x}) \log p(i|\boldsymbol{x})$, we have the gradient of ℓ

with respect to $\boldsymbol{z}$ as follows:

$$\begin{aligned}\frac{\partial \ell}{\partial \boldsymbol{z}} &= -\frac{\partial \log p(y|\boldsymbol{x})}{\partial \boldsymbol{z}} - \lambda \frac{\sum_{i=1}^{K} p(i|\boldsymbol{x}) \log p(i|\boldsymbol{x})}{\partial \boldsymbol{z}} \\ &= -\frac{1}{p(y|\boldsymbol{x})} \cdot \frac{\partial p(y|\boldsymbol{x})}{\partial \boldsymbol{z}} - \lambda \sum_{i=1}^{K} \left(\frac{\partial p(i|\boldsymbol{x})}{\partial \boldsymbol{z}} \cdot \log p(i|\boldsymbol{x}) + \frac{\partial p(i|\boldsymbol{x})}{\partial \boldsymbol{z}} \right) \\ &= \underbrace{-\Big(\frac{1}{p(y|\boldsymbol{x})} + \lambda + \lambda \log p(y|\boldsymbol{x}) \Big) \cdot \frac{\partial p(y|\boldsymbol{x})}{\partial \boldsymbol{z}}}_{\text{active term}} + \underbrace{\lambda \sum_{i \neq y} \Big(-1 - \log p(i|\boldsymbol{x}) \Big) \cdot \frac{\partial p(i|\boldsymbol{x})}{\partial \boldsymbol{z}}}_{\text{passive term}}\end{aligned}$$

Compared to the gradient of CE that only contains one active term (i.e. $-\frac{1}{p(y|\boldsymbol{x})} \cdot \frac{\partial p(y|\boldsymbol{x})}{\partial \boldsymbol{z}}$), our gradient consists of two terms: (1) active term denotes the gradient of learning towards the target y; (2) passive term limits the increase of $p(i|\boldsymbol{x}), \forall i \neq y$, which passively maximize $p(y|\boldsymbol{x})$. As can be observed, when $p(y|\boldsymbol{x}) > \frac{1}{e}$, our active term is enhanced by adding $\lambda + \lambda \log p(y|\boldsymbol{x})$ compared to CE's. When $p(y|\boldsymbol{x}) < \frac{1}{e}$, our passive term is activated to explicitly minimize the $p(i|\boldsymbol{x}), \forall i \neq y$, resulting in the increasing of $p(y|\boldsymbol{x})$. In addition, the passive term is dynamic based on magnitude of $p(y|\boldsymbol{x})$, which provides a error tolerant when label is wrong.

In summary, the proposed method addresses the underfitting problem by the joint of active and passive term in our gradient, which coincides with the core idea of Active Passive Loss (Ma *et al.*, 2020).

5.5 Experiments

In this section, we first briefly describe the label noise generation and training details for a fair comparison with existing methods. Then we provide some empirical analyses towards a better understanding of the proposed approach. Finally, we investigate the

effectiveness of the proposed approach by comparing it with state-of-the-art methods on simulated label noise, including MNIST (LeCun *et al.*, 1998), CIFAR-10 and CIFAR-100 (Krizhevsky *et al.*, 2009), and a real-world noisy dataset Webvision (Li *et al.*, 2017). All the experiment implementations are based on PyTorch and run on a NVIDIA A100 GPU.

Label Noise Generation. We simulate the noisy labels following (Patrini *et al.*, 2017). Suppose the label noise rate is η, symmetric label noise is generated by uniformly flipping the label to one of the other class label by given probability $\frac{\eta}{K-1}$. Asymmetric label noise is generated by flipping the original label to a similar class label to mimic the real-world label noise, where the label errors only occur within very similar classes. Specifically, for MNIST, flipping $2 \rightarrow 7$, $3 \rightarrow 8$, $5 \rightarrow 6$ and $7 \rightarrow 2$ with probability η. For CIFAR-10, mapping $truck \rightarrow automobile$, $bird \rightarrow airplane$, $deer \rightarrow horse$ and $cat \leftrightarrow dog$. For CIFAR-100, the 100 classes are grouped into 20 super-classes with each having 5 sub-classes. The asymmetric noise flips each sub-class into the next, circularly within super-classes.

Baselines. In our experiments, we consider four state-of-the-art robust loss functions: (1) Generalized Cross Entropy (GCE) (Zhang and Sabuncu, 2018): $\frac{1-p(y|\boldsymbol{x})^{\rho}}{\rho}$ where $\rho \in (0, 1]$; (2) Symmetric Cross Entropy (SCE) (Wang *et al.*, 2019): α·CE+β·RCE; (3) Taylor Cross Entropy (TCE) (Feng *et al.*, 2021): $\sum_{i=1}^{t} \frac{(1-p(y|\boldsymbol{x}))^i}{i}$; (4) Active Passive Loss (APL) (Ma *et al.*, 2020): $\alpha \cdot \mathrm{NCE} + \beta \cdot \mathrm{MAE}$; We also train networks using CE and FL. For the proposed method, we add the entropy term to CE, FL and GCE, i.e., CE+EM, FL+EM and GCE+EM.

Networks and Training Details. For a fair comparison, we use a 4-layer CNN for MNIST, an 8-layer CNN for CIFAR-10 and a ResNet-34 for CIFAR-100 following (Ma

et al., 2020). The networks are trained for 50, 120, 200 epochs for MNIST, CIFAR-10, CIFAR-100, respectively. For all the training, we use SGD with momentum 0.9 and cosine annealing learning rate scheduler. The weight decay is set as 1×10^{-3}, 1×10^{-4} and 1×10^{-5} for MNIST, CIFAR-10, CIFAR-100 respectively. The initial learning rate is set to 0.01 for MNIST/CIFAR-10 and 0.1 for CIFAR-100. Batch size is set to 128. Typical data augmentations including random width/height shift and horizontal flip are applied.

Hyperparameter Setting. We set the parameters which match their original papers for all baseline method. Specifically, for FL, we set $\gamma = 0.3$. For GCE, we set set $\rho = 0.7$. For SCE, we set $A = -4$, and $\alpha = 0.01, \beta = 1$ for MNIST, $\alpha = 0.1, \beta = 1$ for CIFAR-10, $\alpha = 6, \beta = 0.1$ for CIFAR-100. For TCE, we set $t = 2$ for MNIST and CIFAR-10, $t = 6$ for CIFAR-100. For APL, we set $\alpha = 1, \beta = 100$ for MNIST, $\alpha = 1, \beta = 1$ for CIFAR-10, and $\alpha = 10, \beta = 1$ for CIFAR-100 (Note that the original setting in (Ma *et al.*, 2020) is $\alpha = 10, \beta = 0.1$ for CIFAR-100, but we find that set $\alpha = 10, \beta = 1$ achieves much better performance for APL).

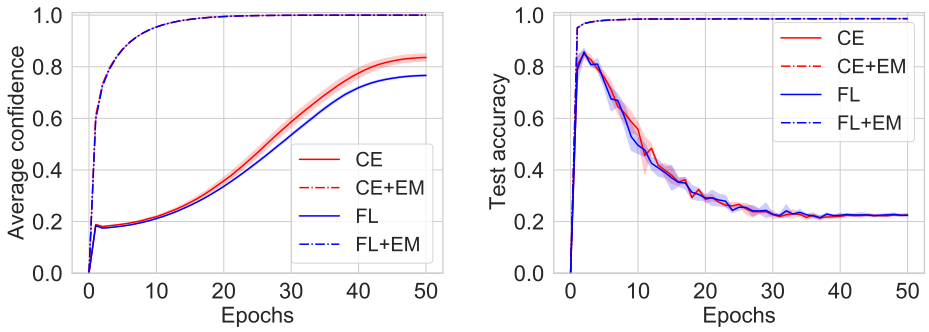
5.5.1 Empirical Understandings

High Confidence Enhances Noise Robustness. To empirically explore the relationship between high confidence and noise robustness, we run a set of experiments on MNIST with 0.8 symmetric label noise. For each training sample, we calculate the confidence of its prediction by $\max_j p(j|\boldsymbol{x})$. Figure 5.3 shows the average confidence and test accuracy in training. For clear comparison, we add the proposed entropy minimization (EM) term to the loss functions that are not robust to noisy label, including CE and FL. As can be observed, both CE and FL have low average confidence

in the early stage, thus the predictions can easily change due to the memorization of mislabeled samples, leading to poor test accuracy. After applying EM, the average confidence maintains a high value after several epochs. The test accuracy of CE+EM and FL+EM show the model strong robustness to label noise. Therefore, maintaining high confident predictions throughout the training does enhance the robustness of model when training data contains noisy labels.

Fitting Ability. Existing noise-robust loss functions, including MAE, NCE, and RCE, have been observed to suffer from an underfitting problem in complex datasets (Zhang and Sabuncu, 2018; Ma *et al.*, 2020). We first run an example to show how these loss functions perform on CIFAR-100 dataset with 0.6 symmetric noise. As can be observed in Figure 5.4, CE and FL are not robust to label noise as there is a clear drop in the test accuracy indicates that DNN starts to memorize mislabeled samples. NCE, MAE and RCE are proven to be robust to label noise. However, these loss functions do not lead to accurate models but attain even worse performance compared to CE and FL. MAE and RCE even fail to converge in this scenario. In contrast, after applying entropy term on non-robust losses, their performance become stable, which demonstrates that EM-enhanced can be both theoretically robust and learning sufficient.

Learned Representations. We investigate the representations learned by CE+EM compared to that learned by CE and APL. Specifically, we extract the high-dimensional representation at the penultimate layer, then project to a 2D embedding using t-SNE (Van der Maaten and Hinton, 2008). Figure 5.6 shows the projected representations for MNIST with different levels of noisy labels respectively. Under all noisy settings, the representations learning by CE+EM are of significantly better quality than these

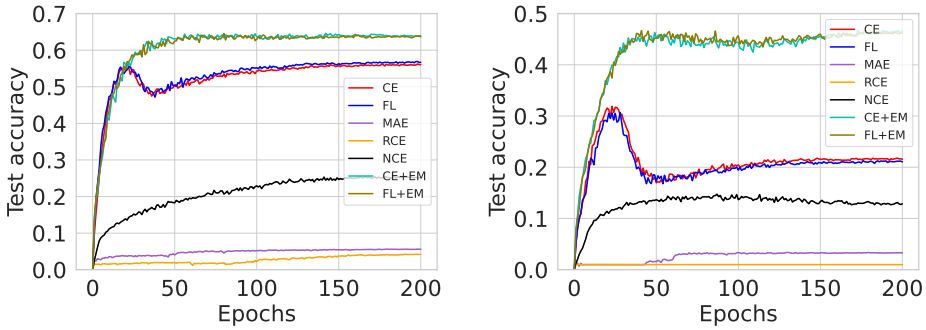


(a) Average Confidence (b) Test Accuracy

Figure 5.3: Average confidence and test accuracy of various approaches on MNIST with 0.8 symmetric label noise. Entropy minimization effectively enhances the robustness of model.

learned by others with more separated and clearly bounded clusters. We can observe that the state-of-the-art method APL has good performance in low levels of label noise (see Figure 5.6(f) when symmetric $\eta = 0.4$). However, its boundary becomes blurred as the noise ratio increases (see Figure 5.6(g) when symmetric noise $\eta = 0.8$). In the asymmetric noise $\eta = 0.4$ case (Figure 5.6(d)), the representations of class 5 (brown) and class 6 (pink) learned by CE are intertwined with each other. Slightly better when using APL. When learned with the proposed method, the representations of class 5 (brown) and class 6 (pink) become more discriminative and the margin is more separated.

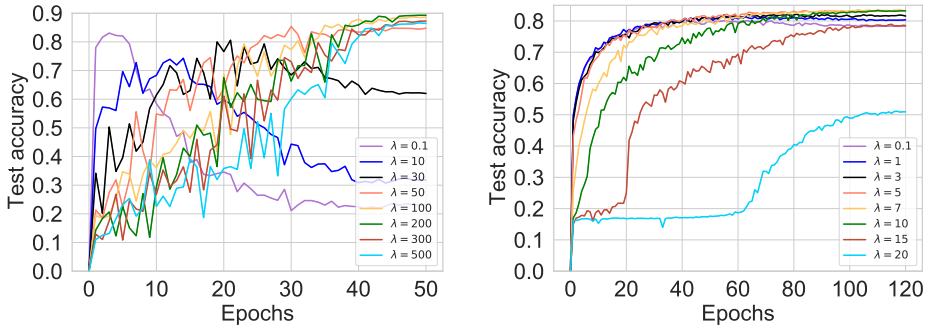
Parameter Analysis. In our method, the hyperparameter $\lambda > 0$ is used to control the strength of entropy minimization. We provide an empirical study to see the effect of different λ in CE+EM. The experiments are conducted on MNIST with 0.8 symmetric label noise and CIFAR-10 with 0.4 symmetric label noise. We test $\lambda \in \{0.1, 10, 30, 50, 100, 200, 300, 500\}$ and $\lambda \in \{0.1, 1, 3, 5, 7, 10, 15, 20\}$ for MNIST



(a) symmetric $\eta = 0.2$ (b) symmetric $\eta = 0.6$

Figure 5.4: Test accuracies of existing noise-robust loss functions versus EM-enhanced loss on CIFAR-100 under 0.2 and 0.6 symmetric noise.

and CIFAR-10 respectively. Figure 2.9 shows the results. As can be observed, the datasets with different complexity require different λ. Besides, λ cannot be too small, otherwise noise-robust cannot be guaranteed (see $\lambda = 0.1, 10, 30$ on MNIST). On the other hand, λ cannot be too large, otherwise the CE loss term will be neglected, resulting in suffering from underfitting problem (see $\lambda = 20$ on CIFAR-10). In our implementation, we gradually increase the value of λ during training. Specifically, we adopt two strategies and their combinations. 1) An exponential way $\lambda = \rho^{\lfloor T/r \rfloor} (\rho \geq 1)$, where T denotes the training epoch and r denotes the updating rate of λ. 2) A linear way $\lambda = aT - b$, where a and b denote the gradient and intercept. For all three EM-enhanced losses, we set $\lambda = 2.5^{T/5}$ for MNIST. Similar strategies for CIFAR-10, CIFAR-100 and Webvision can be found in supplementary material.



(a) MNIST (b) CIFAR-10

Figure 5.5: Test accuracies curve of different λ on MNIST with 0.8 symmetric label noise and CIFAR-10 with 0.4 symmetric label noise.

5.5.2 Evaluation on Benchmark Datasets

The classification accuracies under symmetric and asymmetric label noise are reported in Table 5.2. We observe that the EM-enhanced losses (i.e. CE+EM, FL+EM and GCE+EM) achieve the top 3 best results in most test scenarios across all benchmarks. On clean CIFAR-10 and CIFAR-100, some existing methods, such as GCE and APL, have an observable performance drop due to their limited fitting ability. In contrast, the proposed losses still consistently outperform regular loss functions (i.e. CE and FL), which demonstrates their superior fitting ability. On symmetric label noise, the proposed losses obtain a substantial improvement over the most scenarios. For example, on MNIST with 0.8 symmetric label noise, CE+EM improves the accuracy from 22.49% to 98.69% over CE, better than best baseline APL by 26.23%. On CIFAR-10 with 0.8 symmetric label noise, CE+EM outperforms CE by more than 28%. On CIFAR-100 with 0.8 symmetric label noise where both CE and FL fail to converge, our losses still achieve comparable performance to the state-of-the-art

methods GCE. Overall, the proposed losses are not only robust to label noise, but also guarantee and even improve the fitting ability.

On asymmetric label noise, the proposed losses significantly improve the classification performance across all benchmarks. Even in some noise cases, the performance of our method is not inferior to the performance in the clean case. For example, CE+EM achieves 99.20% on MNIST with 0.4 asymmetric label noise, which is higher than the performance of CE on clean MNIST. On complicated dataset CIFAR-100, our losses also obtain excellent performance. For example, on CIFAR-100 with 0.4 asymmetric label noise, the highest accuracy that can be achieved by existing methods is 42.65% (by APL), which is still 12% lower than our CE+EM and 14% lower than our FL+EM. More experiment results, such as test accuracy vs. epochs, are provided in supplementary materials.

In summary, the EM-enhanced losses provide a consistently strong performance across different datasets with different types of noise, which verifies the effectiveness of entropy minimization for robust learning with noisy labels.

5.5.3 Evaluation on Real-world Noisy Labels

In the above experiments, we have seen that the proposed approach achieves excellent performance on benchmarks with simulated label noise. Here we test the effectiveness of the proposed method on large-scale real-world dataset Webvision (Li *et al.*, 2017). Webvision contains 2.4 millions images crawled from the web (e.g. Flicker and Google) based on 1,000 classes of ImageNet ILSVRC12 (Deng *et al.*, 2009). The label noise level of Webvision is estimated at 20%. Here, we follow the "Mini" setting in (Jiang *et al.*, 2018) that only takes the first 50 classes of the Google resized image subset.

Datasets	Methods	Clean ($\eta = 0.0$)	Symmetric Noise Rate (η)				Asymmetric Noise Rate (η)			
			0.2	0.4	0.6	0.8	0.1	0.2	0.3	0.4
MNIST	CE	99.16 ± 0.09	91.57 ± 0.14	74.37 ± 0.16	50.01 ± 0.48	22.49 ± 0.20	97.68 ± 0.22	94.08 ± 0.18	89.16 ± 0.26	82.21 ± 0.31
	FL	99.15 ± 0.06	91.80 ± 0.07	74.59 ± 0.15	50.37 ± 0.07	22.48 ± 0.59	97.57 ± 0.11	94.40 ± 0.09	88.86 ± 0.30	82.46 ± 0.23
	GCE	99.16 ± 0.02	98.82 ± 0.02	96.92 ± 0.14	81.17 ± 1.02	33.19 ± 0.67	99.09 ± 0.09	96.45 ± 0.14	89.37 ± 0.33	81.61 ± 0.13
	SCE	**99.26 ± 0.02**	98.92 ± 0.02	97.40 ± 0.16	88.65 ± 0.88	48.33 ± 0.08	99.14 ± 0.04	98.04 ± 0.07	93.90 ± 0.14	85.04 ± 0.35
	TCE	99.17 ± 0.05	98.89 ± 0.03	98.18 ± 0.11	91.53 ± 0.78	46.37 ± 0.79	99.10 ± 0.04	98.64 ± 0.08	93.21 ± 0.31	82.01 ± 0.33
	APL	**99.36 ± 0.04**	99.10 ± 0.06	98.58 ± 0.08	95.95 ± 0.06	72.46 ± 0.96	**99.34 ± 0.00**	98.83 ± 0.12	96.90 ± 0.17	91.79 ± 0.92
	CE+EM	99.15 ± 0.06	**99.19 ± 0.03**	**99.08 ± 0.06**	**98.93 ± 0.02**	**98.69 ± 0.02**	**99.30 ± 0.03**	**99.19 ± 0.03**	**99.20 ± 0.04**	**99.20 ± 0.03**
	FL+EM	99.12 ± 0.04	**99.13 ± 0.02**	**99.07 ± 0.05**	**98.98 ± 0.02**	**98.65 ± 0.08**	**99.28 ± 0.04**	**99.27 ± 0.04**	**99.21 ± 0.02**	**99.19 ± 0.01**
	GCE+EM	**99.28 ± 0.03**	**99.21 ± 0.03**	**99.02 ± 0.02**	**98.92 ± 0.05**	**98.64 ± 0.05**	99.25 ± 0.02	**99.26 ± 0.02**	**99.22 ± 0.08**	**99.18 ± 0.07**
CIFAR-10	CE	90.53 ± 0.16	74.91 ± 0.22	57.70 ± 0.53	38.68 ± 0.48	19.38 ± 0.39	87.21 ± 0.23	83.59 ± 0.24	79.47 ± 0.19	75.07 ± 0.35
	FL	89.99 ± 0.29	74.33 ± 0.45	57.69 ± 0.63	39.06 ± 0.30	18.96 ± 0.63	86.76 ± 0.06	83.31 ± 0.15	79.35 ± 0.03	74.53 ± 0.29
	GCE	89.27 ± 0.19	87.22 ± 0.20	82.39 ± 0.24	67.51 ± 0.52	25.50 ± 0.84	88.56 ± 0.14	85.97 ± 0.29	80.58 ± 0.34	74.21 ± 0.58
	SCE	**91.45 ± 0.06**	87.91 ± 0.24	79.28 ± 0.59	61.61 ± 0.57	27.55 ± 0.64	89.94 ± 0.17	86.83 ± 0.18	81.41 ± 0.17	75.30 ± 0.39
	TCE	89.59 ± 0.11	87.38 ± 0.13	84.05 ± 0.27	74.53 ± 0.07	38.19 ± 0.83	88.66 ± 0.14	87.00 ± 0.08	82.80 ± 0.06	66.46 ± 2.20
	APL	88.93 ± 0.05	87.14 ± 0.22	83.81 ± 0.25	75.98 ± 0.39	45.10 ± 0.35	88.17 ± 0.07	86.80 ± 0.06	83.64 ± 0.17	76.96 ± 0.43
	CE+EM	91.25 ± 0.16	**88.16 ± 0.12**	**86.13 ± 0.19**	**79.51 ± 1.07**	**48.34 ± 0.48**	**90.15 ± 0.07**	**89.01 ± 0.10**	**86.87 ± 0.21**	**80.27 ± 0.36**
	FL+EM	**91.27 ± 0.06**	**88.09 ± 0.29**	**86.21 ± 0.18**	**79.85 ± 0.33**	**48.44 ± 0.16**	**90.14 ± 0.46**	**89.38 ± 0.40**	**87.54 ± 0.13**	**83.38 ± 0.67**
	GCE+EM	**91.35 ± 0.05**	**88.10 ± 0.27**	**86.22 ± 0.21**	**80.18 ± 1.12**	**47.21 ± 0.98**	**90.08 ± 0.16**	**89.05 ± 0.05**	**86.64 ± 0.27**	**80.30 ± 0.40**
CIFAR-100	CE	71.44 ± 0.24	55.65 ± 0.54	40.72 ± 0.99	22.14 ± 0.70	7.57 ± 0.22	64.36 ± 1.07	58.43 ± 0.26	50.74 ± 0.23	41.45 ± 0.36
	FL	70.66 ± 0.13	55.52 ± 1.35	40.21 ± 0.21	22.10 ± 0.77	7.25 ± 0.27	64.50 ± 1.11	58.28 ± 0.95	50.75 ± 0.75	40.96 ± 0.50
	GCE	64.39 ± 1.99	61.75 ± 0.79	56.88 ± 1.24	**46.65 ± 0.64**	**18.55 ± 0.83**	61.14 ± 0.49	60.37 ± 0.95	53.50 ± 1.30	40.62 ± 0.92
	SCE	71.04 ± 0.46	55.20 ± 0.60	39.72 ± 0.82	21.14 ± 0.48	7.37 ± 0.27	64.68 ± 0.39	56.74 ± 0.66	50.10 ± 0.33	41.07 ± 0.69
	TCE	30.97 ± 1.29	31.76 ± 1.93	31.91 ± 1.15	29.38 ± 1.18	13.40 ± 0.78	30.25 ± 0.68	29.05 ± 1.25	27.04 ± 0.46	22.24 ± 0.85
	APL	67.65 ± 0.30	63.85 ± 0.31	58.42 ± 0.08	44.88 ± 0.65	**24.08 ± 1.09**	66.22 ± 0.27	62.95 ± 0.63	56.80 ± 0.45	42.65 ± 0.14
	CE+EM	**71.80 ± 0.10**	**64.59 ± 0.22**	**59.56 ± 0.36**	45.17 ± 0.31	**16.92 ± 0.23**	**70.41 ± 0.23**	**68.56 ± 0.16**	**65.54 ± 0.19**	**54.82 ± 1.87**
	FL+EM	**71.76 ± 0.31**	**64.46 ± 0.66**	**59.98 ± 0.37**	**45.32 ± 0.70**	15.82 ± 0.15	**70.75 ± 0.09**	**68.74 ± 0.38**	**66.69 ± 0.37**	**56.76 ± 0.45**
	GCE+EM	**72.05 ± 0.15**	**64.45 ± 0.37**	**60.03 ± 0.06**	**45.66 ± 0.70**	16.78 ± 0.43	**70.37 ± 0.14**	**68.36 ± 0.32**	**65.78 ± 0.54**	**55.84 ± 1.77**

Table 5.2: Test accuracy (%) of different methods on benchmark datasets with clean or symmetric label noise ($\eta \in [0.2, 0.4, 0.6, 0.8]$). The results (mean ± std) are reported over 3 random runs and the top 3 best results are in boldfaced.

Top-1 accuracy / Loss	CE	FL	GCE	SCE	APL	CE+EM	FL+EM
On Webvision validation	62.32	63.16	56.12	65.16	64.00	**67.56**	**66.56**
On ILSVRC12 validation	57.56	58.12	55.28	60.56	59.36	**62.28**	**62.48**

Table 5.3: Top-1 validation accuracies (%) on both Webvision and ILSVRC12 validation set of ResNet-50 models trained on WebVision using different loss functions, under the Mini setting in (Jiang *et al.*, 2018).

Besides th noisy labels, Webvision is also an imbalanced dataset with an imbalanced factor (i.e. the number of training samples in the largest class divided by the smallest) equal to 6.78. We follow the settings in APL (Ma *et al.*, 2020) for fair comparison with existing methods. All the models are evaluated on the clean ILSVRC12 and WebVision validation set. For each loss function, we train a ResNet-50 (He *et al.*, 2016a) using SGD for 250 epochs with initial learning rate 0.4, nesterov momentum 0.9 and weight decay 3×10^{-5} and batch size 256. The learning rate is multiplied by 0.97 after every epoch of training. We resize the images to 224×224. Typical

data augmentations including random width/height shift, color jittering and random horizontal flip are applied. We implement these methods based on public code and use the same hyperparameters as suggested in their original papers. For GCE, we set $q = 0.7$, while for SCE, we use the setting with $A = -4$, $\alpha = 10$ and $\beta = 1$. For APL (NCE + MAE), we set $\alpha = 50$ and $\beta = 1$. The top-1 validation accuracies of different loss functions on the WebVision and ILSVRC12 validation set are reported in Table 5.3. As can be observed, both the proposed losses outperform existing loss functions by a clear margin, which validates the effectiveness of our method against real-world label noise.

5.6 Conclusion

In this paper, we observe the fluctuation of model output in learning with noisy labels and propose to stabilize the model output for consistent predictions during training. We revealed a new insight into noise-robust loss functions that: entropy minimization can make non-robust loss functions robust to label noise without suffering from the underfitting problem, outperforming most existing robust loss functions such as MAE, RCE and NCE. We provide both empirical and theoretical analyses to verify the effectiveness of proposed method. Moreover, our experimental results demonstrate that the proposed method achieves superior performance over the state-of-the-art losses on both simulated and real-world datasets.

5.7 Appendix

5.7.1 Complete Proof of Theorems

Theorem 5.7.1. *In a multi-class classification problem, ℓ is noise tolerant under symmetric label noise if noise rate $\eta < \frac{K-1}{K}$ and $f : \mathcal{X} \to V$. And, given $R_\ell(f^*) = 0$ and $0 \leq \ell(f^*(\boldsymbol{x}), k) \leq \frac{C}{K-1}$, ℓ is also noise tolerant under asymmetric label noise or class-dependent label noise when noise rate $\eta_{yk} < 1 - \eta_y$ with $\sum_{k \neq y} \eta_{yk} = \eta_y$.*

Proof. For symmetric noise with any $f \in \mathcal{H}_v$, we can derive

$$\begin{aligned} R_\ell^\eta(f) &= \mathbb{E}_{\hat{D}}\Big[\ell(f(\boldsymbol{x}), \hat{y})\Big] \\ &= \mathbb{E}_D\Big[(1-\eta)\ell(f(\boldsymbol{x}), y) + \frac{\eta}{K-1}\sum_{j \neq y}\ell(f(\boldsymbol{x}), j)\Big] \\ &= (1-\eta)R_l(f) + \frac{\eta}{K-1}\Big(\sum_{j=1}^{K}\ell(f(\boldsymbol{x}), j) - R_\ell(f)\Big) \\ &= \big(1 - \frac{\eta K}{K-1}\big)R_\ell(f) + \frac{\eta}{K-1}\sum_{j=1}^{K}\ell(f(\boldsymbol{x}), j) \\ &= \big(1 - \frac{\eta K}{K-1}\big)R_\ell(f) + \frac{\eta C}{K-1}. \end{aligned} \tag{5.18}$$

Thus, we have

$$R_\ell^\eta(f^*) - R_\ell^\eta(f_\eta^*) = \big(1 - \frac{\eta K}{K-1}\big)(R_\ell(f^*) - R_\ell(f_\eta^*)) \leq 0, \tag{5.19}$$

where f^* and f_η^* is a minimizer of R_ℓ and R_ℓ^η. Since $1 - \frac{\eta K}{K-1} > 0$. and $R_\ell(f^*) - R_\ell(f_\eta^*) \leq 0$, we have $R_\ell^\eta(f^*) - R_\ell^\eta(f_\eta^*) \leq 0$, which proves f^* is also a minimizer of R_ℓ^η. For

asymmetric noise, we have

$$\begin{aligned} R_\ell^\eta(f) &= \mathbb{E}_{\hat{D}}\Big[\ell(f(\boldsymbol{x}), \hat{y})\Big] \\ &= \mathbb{E}_D\Big[(1-\eta_y)\ell(f(\boldsymbol{x}), y) + \sum_{k\neq y}\eta_{yk}\ell(f(\boldsymbol{x}), k)\Big] \\ &= \mathbb{E}_D\Big[(1-\eta_y)\Big(\sum_{k=1}^{K}\ell(f(\boldsymbol{x}), k) - \sum_{k\neq y}\ell(f(\boldsymbol{x}), k)\Big)\Big] + \mathbb{E}_D\Big[\sum_{k\neq y}\eta_{yk}\ell(f(\boldsymbol{x}), k)\Big] \\ &= C\mathbb{E}_D(1-\eta_y) - \mathbb{E}_D\Big[\sum_{k\neq y}(1-\eta_y-\eta_{yk})\ell(f(\boldsymbol{x}), k)\Big]. \end{aligned} \tag{5.20}$$

Let f_η^* and f^* be the minimizer of R_ℓ^η and R_ℓ when $f : \mathcal{X} \to V$, respectively. We have $R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) \leq 0$ and hence derive that

$$\mathbb{E}_D\Big[\sum_{k\neq y}(1-\eta_y-\eta_{yk})(\ell(f^*(\boldsymbol{x}), k) - \ell(f_\eta^*(\boldsymbol{x}), k))\Big] \leq 0 \tag{5.21}$$

First $(1-\eta_y-\eta_{yk}) > 0$ as per the assumption $\eta_{yk} < 1-\eta_y$, we have $\ell(f^*(\boldsymbol{x}), k) \leq \ell(f_\eta^*(\boldsymbol{x}), k)$. Second, our assumption has $R_\ell(f^*) = 0$, we have $\ell(f^*(\boldsymbol{x}), y) = 0$. Since the condition $0 \leq \ell(f^*(\boldsymbol{x}), k) \leq \frac{C}{K-1}$, we have $\ell(f^*(\boldsymbol{x}), k) = \frac{C}{K-1}$ for all $k \neq y$. Then we have $\ell(f_\eta^*(\boldsymbol{x}), k) \geq \frac{C}{K-1}$. For both conditions hold, we have $\ell(f_\eta^*(\boldsymbol{x}), k) = \frac{C}{K-1}$, which implies $\ell(f_\eta^*(\boldsymbol{x}), y) = 0$. Therefore, $\ell(f^*(\boldsymbol{x}), k) = \ell(f_\eta^*(\boldsymbol{x}), k)$ for all $k \in [K]$, thus $f_\eta^* = f^*$ which completes the proof. □

Theorem 5.7.2. *In a multi-class classification problem, let $f : \mathcal{X} \to O$. Suppose the loss ℓ satisfies $|\sum_{k=1}^{K}(\ell(f(\boldsymbol{x})_1, k) - \ell(f(\boldsymbol{x})_2, k))| \leq \delta$ when $|f(\boldsymbol{x})_1 - f(\boldsymbol{x})_2| \leq \epsilon$, and $\delta \to 0$ when $\epsilon \to 0$. Then for symmetric label noise satisfying $\eta < \frac{K-1}{K}$, the risk bound can be expressed as*

$$R_\ell(f_\eta^*) - R_\ell(f^*) \leq \frac{2\eta\delta}{K-1-\eta K} \tag{5.22}$$

where f_η^* and f^* denote the minimizer of $R_\ell^\eta(f)$ and $R_\ell(f)$ when $f \in \mathcal{H}_o$, respectively.

Proof. For symmetric label noise, we have

$$\begin{aligned}
R_\ell^\eta(f^*) &= \mathbb{E}_{\hat{D}}\Big[\ell(f^*(\boldsymbol{x}), \hat{y})\Big] \\
&= \mathbb{E}_D\Big[(1-\eta)\ell(f^*(\boldsymbol{x}), y) + \frac{\eta}{K-1}\sum_{k\neq y}\ell(f^*(\boldsymbol{x}), k)\Big] \\
&= (1-\eta)R_l(f^*) + \frac{\eta}{K-1}\Big(\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) - R_\ell(f^*)\Big) \\
&= (1-\frac{\eta K}{K-1})R_\ell(f^*) + \frac{\eta}{K-1}\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) \\
&= (1-\frac{\eta K}{K-1})R_\ell(f^*) + \frac{\eta}{K-1}\sum_{k=1}^{K}\ell(v, k) + \frac{\eta}{K-1}\delta_1,
\end{aligned} \tag{5.23}$$

where $\delta_1 = \mathbb{E}_D\Big[\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) - \sum_{k=1}^{K}\ell(v, k)\Big]$. Given $f^* \in \mathcal{H}_o$, we have $|f^*(\boldsymbol{x}) - v| \leq \epsilon$ and $|\sum_{k=1}^{K}\ell(f^*(\boldsymbol{x}), k) - \sum_{k=1}^{K}\ell(v, k)| \leq \delta$, which implies $\delta_1 \in [-\delta, \delta]$. Similarly, we can obtain

$$R_\ell^\eta(f_\eta^*) = (1-\frac{\eta K}{K-1})R_\ell(f_\eta^*) + \frac{\eta}{K-1}\sum_{k=1}^{K}\ell(v, k) + \frac{\eta}{K-1}\delta_2. \tag{5.24}$$

Based on Eq. (5.23) and Eq. (5.24), we have

$$R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) = (1-\frac{\eta K}{K-1})(R_\ell(f_\eta^*) - R_\ell(f^*)) + \frac{\eta}{K-1}(\delta_2 - \delta_1). \tag{5.25}$$

Given f_η^* and f^* are the minimizers of $R_\ell^\eta(f)$ and $R_\ell(f)$, we have $R_\ell^\eta(f_\eta^*) - R_\ell^\eta(f^*) \leq 0$. As $1 - \frac{\eta K}{K-1} > 0$, we have

$$R_\ell(f_\eta^*) - R_\ell(f^*) \leq \frac{(\delta_2 - \delta_1)\eta}{K - 1 - \eta K} \leq \frac{2\eta\delta}{K - 1 - \eta K} \tag{5.26}$$

□

5.7.2 hyperparameter selection

In our implementation, we gradually increase the value of λ during training. Specifically, we adopt two strategies and their combinations. 1) An exponential way $\lambda = \rho^{\lfloor T/r \rfloor}$, where T denote the training epoch and r denotes the updating rate of λ. 2) A linear way $\lambda = aT - b$, where a and b denote the gradient and intercept. For example, we set $\lambda = 2.5^{T/5}$ for MNIST, $\lambda = 1.03^T$ for CIFAR-10. For CIFAR-100, since it is a more complicated dataset compared to MNIST and CIFAR-100, we combine two strategies and set $\lambda = 1.09^T$ when $T \leq 70$ and $\lambda = T$ when $T > 70$. For Webvision, we set $\lambda = 1.05^T$ when $T \leq 70$ and $\lambda = T - 40$ when $T > 70$.

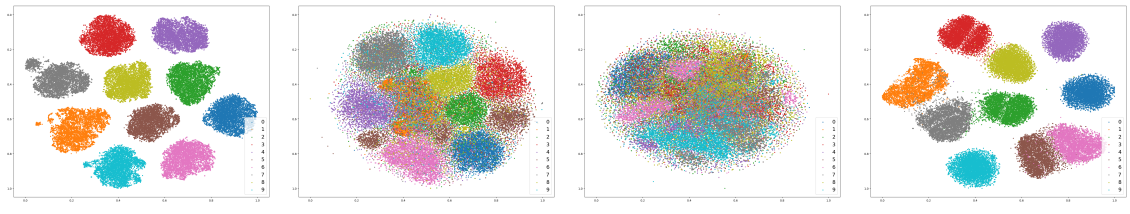
5.7.3 More results on Comparison With state-of-the-art methods

Fig. 5.7 shows test accuracy vs. epochs on MNIST. As can be observed, the regular loss functions CE and FL suffer from significant overfitting in all noisy cases. The existing noise robust loss functions GCE, SCE, TCE and APL perform well under low levels of label noise, but fail to provide robustness under high levels of label noise. In contrast, the proposed losses CE+EM, FL+EM and GCE+EM provides excellent

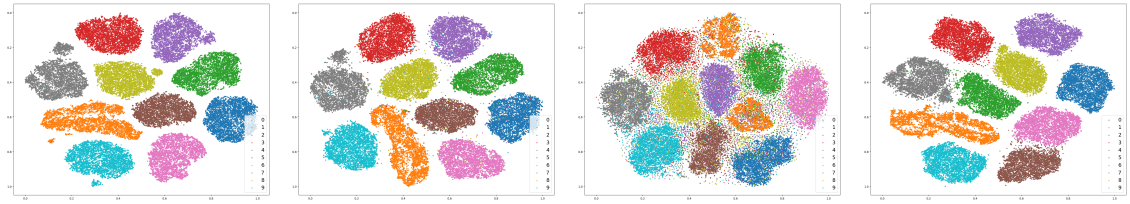
performance consistently. Fig. 5.8 and Fig. 5.9 show the test accuracy vs. epochs on CIFAR-10 and CIFAR-100, respectively. The results are similar to MNIST, our proposed losses achieve the best accuracy in most cases, especially when datasets contain asymmetric label noise.

5.7.4 More results on Learned Representations

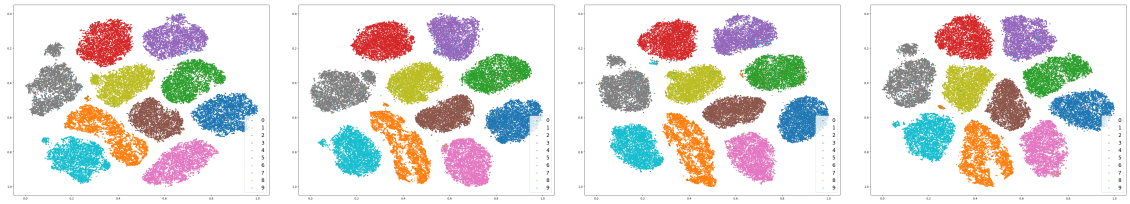
More visualizations of representations learned by other methods on MNIST and CIFAR-10 are shown in Fig. 5.10 and Fig. 5.11. As can be observed, the representations learned by the proposed EM-enhanced losses are more discriminative than those learned by other losses, which are with more separated and clearly bound margins.



(a) CE (clean $\eta = 0.0$) (b) CE (sym $\eta = 0.4$) (c) CE (sym $\eta = 0.8$) (d) CE (asym $\eta = 0.4$)

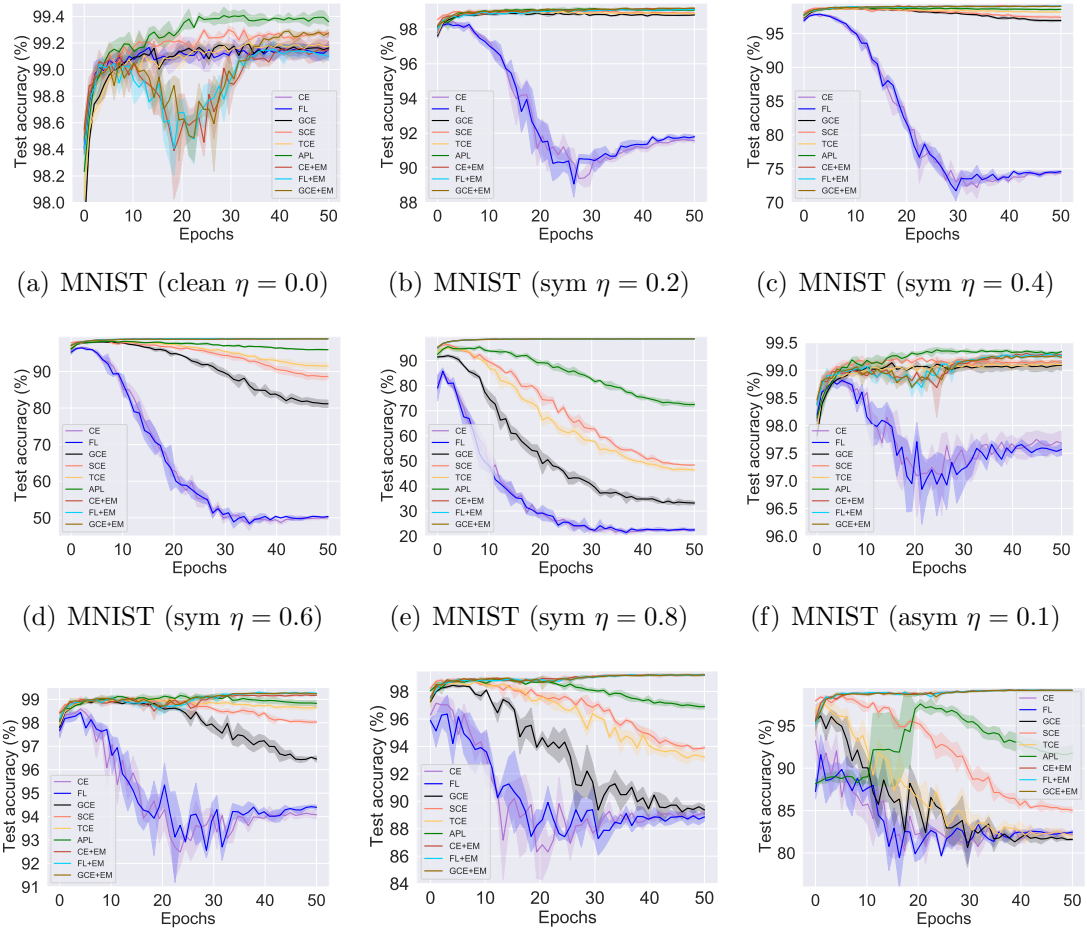


(e) APL (clean $\eta = 0.0$) (f) APL (sym $\eta = 0.4$) (g) APL (sym $\eta = 0.8$) (h) APL (asym $\eta = 0.4$)



(i) CE+EM (clean $\eta = 0.0$) (j) CE+EM (sym $\eta = 0.4$) (k) CE+EM (sym $\eta = 0.8$) (l) CE+EM (asym $\eta = 0.4$)

Figure 5.6: t-SNE (Van der Maaten and Hinton, 2008) plots of feature representations learned by CE, APL, and the proposed CE+EM on MNIST with different ratios of label noise. Different colors represent the different classes. More visualizations of learned representations for other methods and datasets are provided in the supplementary materials.

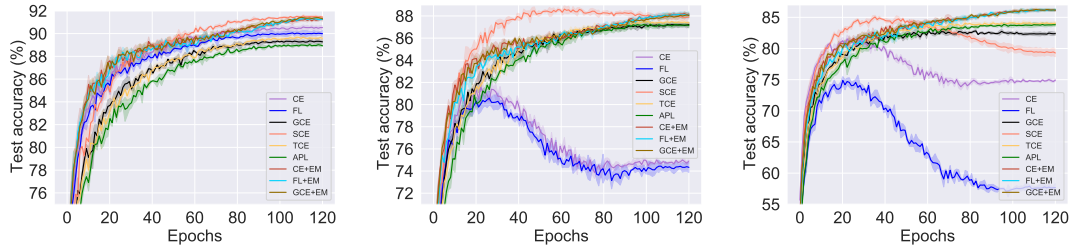


(a) MNIST (clean $\eta = 0.0$) (b) MNIST (sym $\eta = 0.2$) (c) MNIST (sym $\eta = 0.4$)

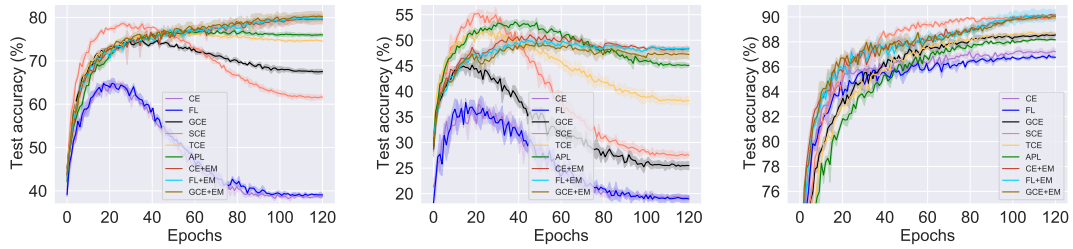
(d) MNIST (sym $\eta = 0.6$) (e) MNIST (sym $\eta = 0.8$) (f) MNIST (asym $\eta = 0.1$)

(g) MNIST (asym $\eta = 0.2$) (h) MNIST (asym $\eta = 0.3$) (i) MNIST (asym $\eta = 0.4$)

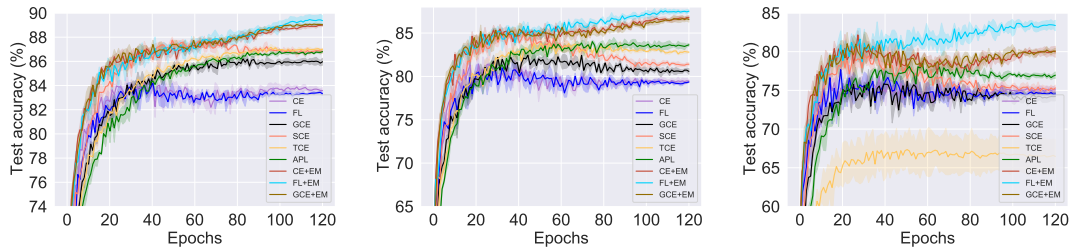
Figure 5.7: Test accuracies of different methods on MNIST with different label noise, where (a) denotes the clean case, (b-e) denote the symmetric label noise, and (f-i) denote the asymmetric label noise.



(a) CIFAR-10 (clean $\eta = 0.0$) (b) CIFAR-10 (sym $\eta = 0.2$) (c) CIFAR-10 (sym $\eta = 0.4$)

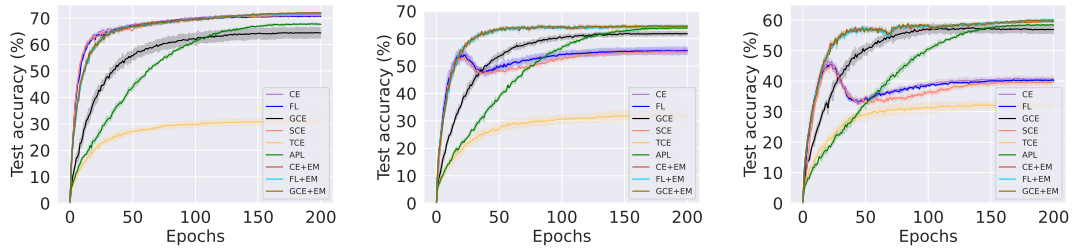


(d) CIFAR-10 (sym $\eta = 0.6$) (e) CIFAR-10 (sym $\eta = 0.8$) (f) CIFAR-10 (asym $\eta = 0.1$)

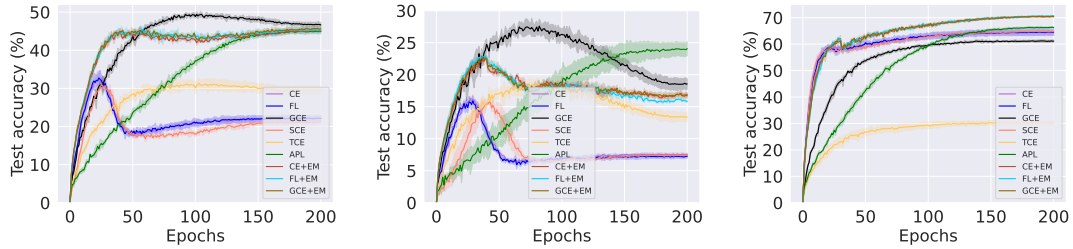


(g) CIFAR-10 (asym $\eta = 0.2$) (h) CIFAR-10 (asym $\eta = 0.3$) (i) CIFAR-10 (asym $\eta = 0.4$)

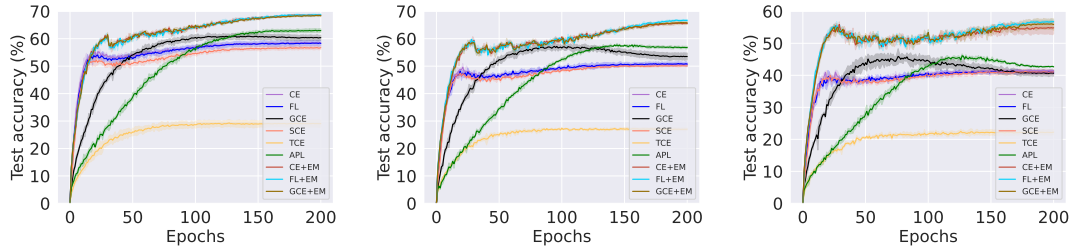
Figure 5.8: Test accuracies of different methods on CIFAR-10 with different label noise, where (a) denotes the clean case, (b-e) denote the symmetric label noise, and (f-i) denote the asymmetric label noise.



(a) CIFAR-100 (clean $\eta = 0.0$) (b) CIFAR-100 (sym $\eta = 0.2$) (c) CIFAR-100 (sym $\eta = 0.4$)

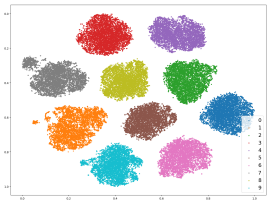


(d) CIFAR-100 (sym $\eta = 0.6$) (e) CIFAR-100 (sym $\eta = 0.8$) (f) CIFAR-100 (asym $\eta = 0.1$)

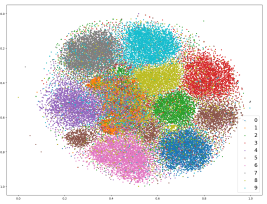


(g) CIFAR-100 (asym $\eta = 0.2$) (h) CIFAR-100 (asym $\eta = 0.3$) (i) CIFAR-100 (asym $\eta = 0.4$)

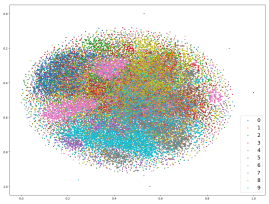
Figure 5.9: Test accuracies of different methods on CIFAR-100 with different label noise, where (a) denotes the clean case, (b-e) denote the symmetric label noise, and (f-i) denote the asymmetric label noise.



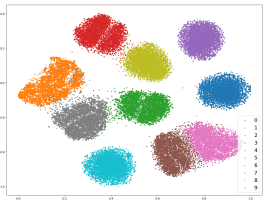
(a) CE (clean $\eta = 0.0$)



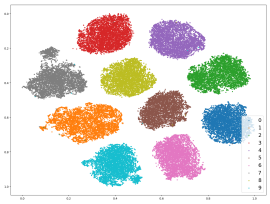
(b) CE (sym $\eta = 0.4$)



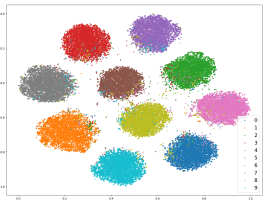
(c) CE (sym $\eta = 0.8$)



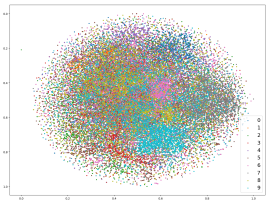
(d) CE (asym $\eta = 0.4$)



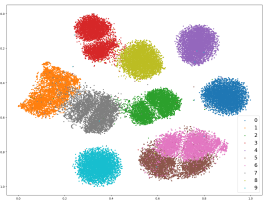
(e) GCE (clean $\eta = 0.0$)



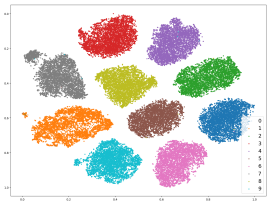
(f) GCE (sym $\eta = 0.4$)



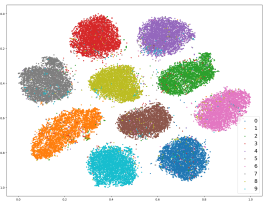
(g) GCE (sym $\eta = 0.8$)



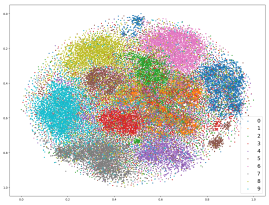
(h) GCE (asym $\eta = 0.4$)



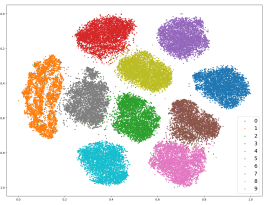
(i) SCE (clean $\eta = 0.0$)



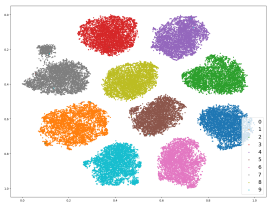
(j) SCE (sym $\eta = 0.4$)



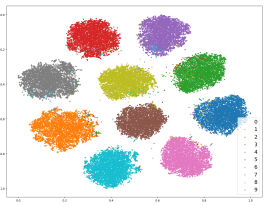
(k) SCE (sym $\eta = 0.8$)



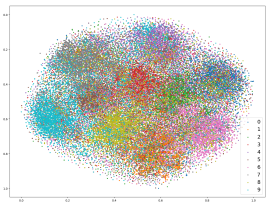
(l) SCE (asym $\eta = 0.4$)



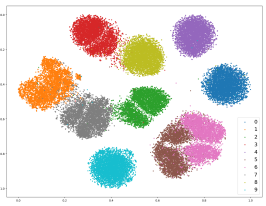
(m) TCE (clean $\eta = 0.0$)



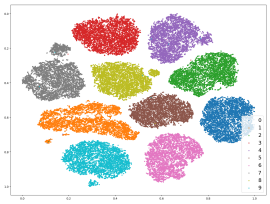
(n) TCE (sym $\eta = 0.4$)



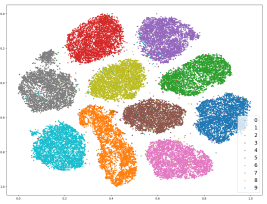
(o) TCE (sym $\eta = 0.8$)



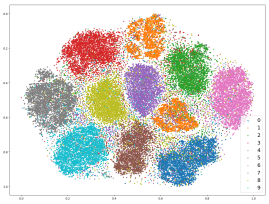
(p) TCE (asym $\eta = 0.4$)



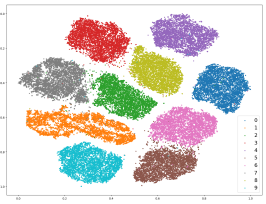
(q) APL (clean $\eta = 0.0$)



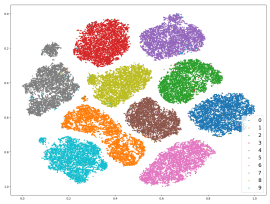
(r) APL (sym $\eta = 0.4$)



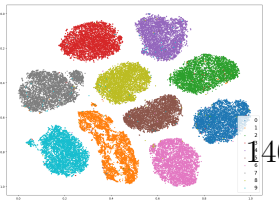
(s) APL (sym $\eta = 0.8$)



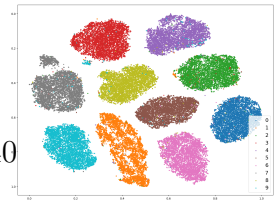
(t) APL (asym $\eta = 0.4$)



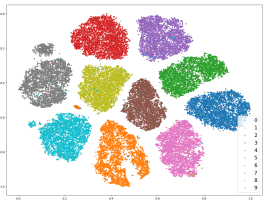
(u) CE+EM (clean $\eta = 0.0$)



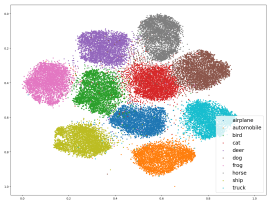
(v) CE+EM (sym $\eta = 0.4$)



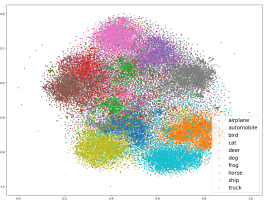
(w) CE+EM (sym $\eta = 0.8$)



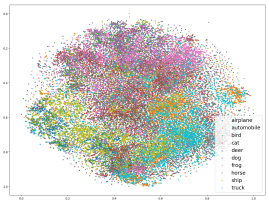
(x) CE+EM (asym $\eta = 0.4$)



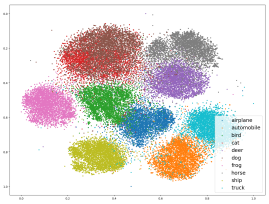
(a) CE (clean $\eta = 0.0$)



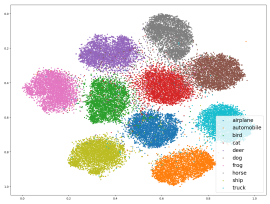
(b) CE (sym $\eta = 0.4$)



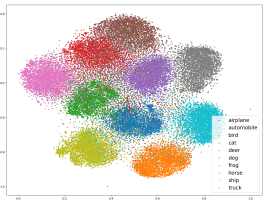
(c) CE (sym $\eta = 0.8$)



(d) CE (asym $\eta = 0.4$)

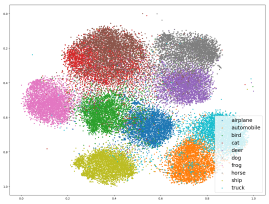


(e) GCE (clean $\eta = 0.0$)

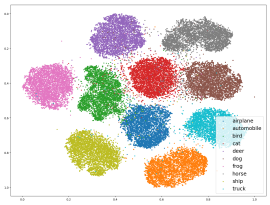


(f) GCE (sym $\eta = 0.4$)

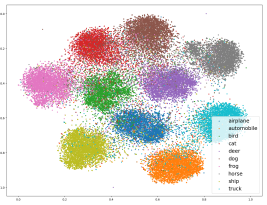
(g) GCE (sym $\eta = 0.8$)



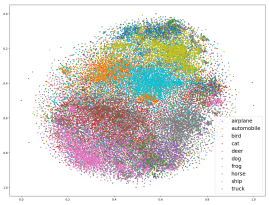
(h) GCE (asym $\eta = 0.4$)



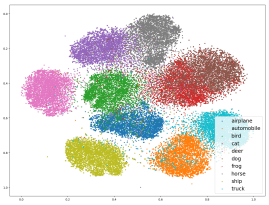
(i) SCE (clean $\eta = 0.0$)



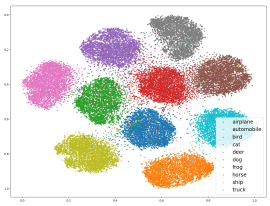
(j) SCE (sym $\eta = 0.4$)



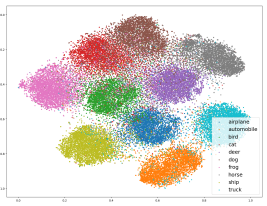
(k) SCE (sym $\eta = 0.8$)



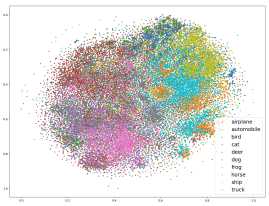
(l) SCE (asym $\eta = 0.4$)



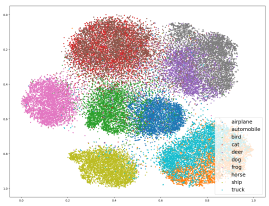
(m) TCE (clean $\eta = 0.0$)



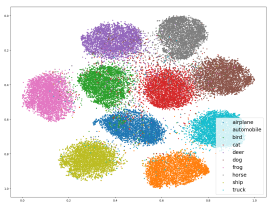
(n) TCE (sym $\eta = 0.4$)



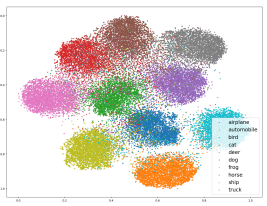
(o) TCE (sym $\eta = 0.8$)



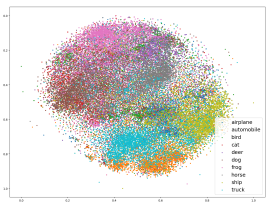
(p) TCE (asym $\eta = 0.4$)



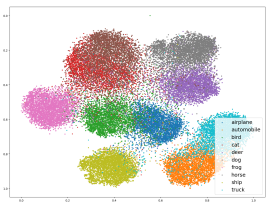
(q) APL (clean $\eta = 0.0$)



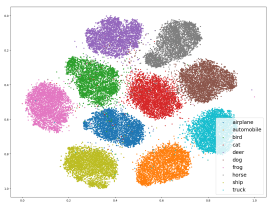
(r) APL (sym $\eta = 0.4$)



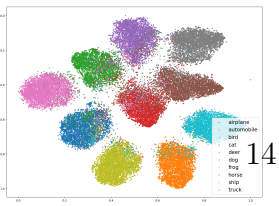
(s) APL (sym $\eta = 0.8$)



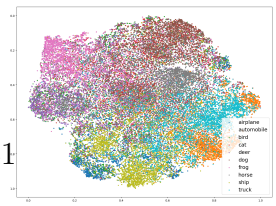
(t) APL (asym $\eta = 0.4$)



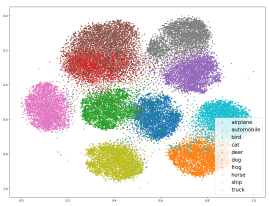
(u) CE+EM (clean $\eta = 0.0$)



(v) CE+EM (sym $\eta = 0.4$)



(w) CE+EM (sym $\eta = 0.8$)



(x) CE+EM (asym $\eta = 0.4$)

Chapter 6

MixNN: Combating Noisy Labels in Deep Learning by Mixing with Nearest Neighbors

6.1 Citation and Main Contributor

Lu, Yangdi, and He, Wenbo. "MixNN: Combating Noisy Labels in Deep Learning by Mixing with Nearest Neighbors." In 2021 IEEE International Conference on Big Data (Big Data), pp. 847-856, doi: 10.1109/BigData52589.2021.9671816.

The main contributor to this paper is the first author - Yangdi Lu (contributes more than 80%).

6.2 Copyright

6.3 Introduction

Deep neural networks (DNNs) have achieved remarkable performance in a variety of applications (e.g. image classification and object detection). Despite the use of novel network architectures and efficient optimization algorithms, large-scale training data with correct labels is always required for these supervised tasks. However, obtaining such high-quality training data with human-annotated labels is extremely expensive and time-consuming in practice. Some non-expert sources, such as Amazon's Mechanical Turk and online searching engines, have been widely used to lower the high labeling cost. However, due to the limited knowledge and inadvertent mistakes, crowdsourced annotators cannot annotate specific tasks with 100% accuracy, resulting in introducing *noisy labels*. Even the most celebrated and highly-curated datasets, such as ImageNet (Deng *et al.*, 2009), are famously containing noisy labels.

Unlike traditional supervised learning, which assumes that all label information is correct, we consider the training data contains a certain percentage of samples with incorrect labels. Training DNNs on such unreliable datasets is known to be highly affected as the significant number of model parameters render DNNs even overfit to noisy labels (Li *et al.*, 2020b). Zhang *et al.* (2018a) have empirically demonstrated that DNNs can easily fit an entire training dataset with any percentage of corrupted labels, and result in poor generalization capacity on a clean test set. Zhu and Wu (2004) have observed that the performance drop caused by label noise is more substantial than by other noises, such as feature noise. Therefore, it is crucial to develop learning algorithms that achieve superior generalization capability in the presence of noisy labels.

Given a training set consisting of clean samples and mislabeled samples, a common

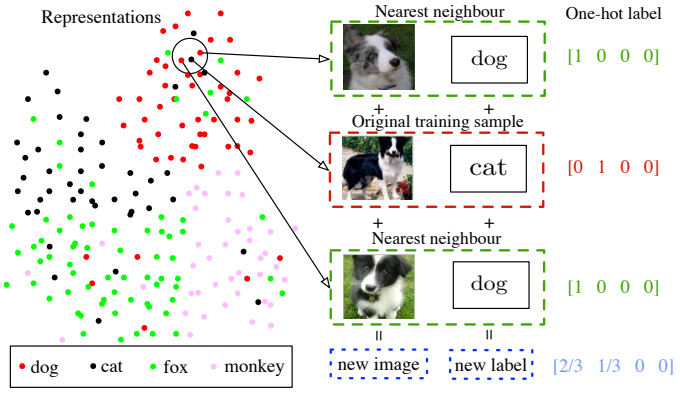


Figure 6.1: The Synthetic samples are generated by mixing itself with its nearest neighbours based on the representations.

approach to mitigate the negative impact of noisy labels is to detect the mislabeled samples and eliminate them in the first stage, then train a new classifier with the remaining clean samples in the second stage (Thulasidasan *et al.*, 2019a; Kim *et al.*, 2019; Huang *et al.*, 2019). However, the first stage's filter mechanism for distinguishing the mislabeled samples from the others is critical to the second stage's classification performance. If the filtering mechanism only removes a few mislabeled samples, the unfiltered mislabeled samples still affect the (supervised) loss and deteriorate the classification performance. On the other hand, if too many samples, including clean samples, are eliminated, the remaining data may not be rich enough to generalize to held-out data effectively in the second stage. Therefore, is it possible to robustly train DNNs on noisy data without discording the informative training samples?

In this paper, we propose an algorithm for combating the negative influence of noisy labels. Our main idea is to generate new synthetic samples that effectively hide the information from the noisy labels, allowing robustly training the DNNs. Specifically, for each training sample, we search its K nearest neighbours based on the

learned representations. Then we linearly combine the samples with their nearest neighbours in terms of their images and labels to create synthetic samples. By training DNNs with these synthetic samples, our approach effectively reduces the noisy supervision from mislabeled samples and enhances the task performance. Take the case in Fig. 6.1 as an example – Suppose we simply average the three images and their labels, then the new image is a mixed dog image and the new label is a smoothed label with the probability of 2/3 for dog class and 1/3 for cat class. It is clear to observe that training DNNs with the synthetic samples is more reasonable than train them with the original ones (e.g. a dog image with a wrong label cat).

Instead of simply averaging the samples with its nearest neighbours, we assign dynamic weights to these selected samples and convexly combine them to generate the synthetic samples. Ideally, the weights for clean samples should be large to preserve the correct information while small for mislabeled samples to suppress the wrong supervision. Previous work (Arpit *et al.*, 2017) has observed that DNNs learn the clean pattern before memorizing the complex noisy pattern during training. Specifically, DNNs learn from clean samples at ease and receive inconsistent error supervision from the wrong samples before over-fitting to the entire dataset. Therefore, the predictions and given labels are likely to be consistent on clean samples and inconsistent on wrong samples, resulting in the separation of their loss values in the early learning phase. Based on this observation, we propose to estimate the weights by fitting a two-component Gaussian Mixture Model (Permuter *et al.*, 2006) to the per-sample loss distribution and calculate the posterior probability of the loss value to measure whether a sample is clean or not, allowing the mixing strategy to be dynamic in training.

To enhance the robustness of our approach, we propose a strategy to estimate target distribution based on model predictions and given noisy labels. The original noisy labels in our loss function are substituted with new soft labels, which has already been proven to effectively improve the generalization in knowledge distillation. In our scenario, the label quality is improved as the wrong labels have been gradually corrected. Subsequently, the labels of synthetic samples become more accurate, providing a more stable supervisory signal during training.

Our main contributions are summarized as follows:

- We provide insights into learned representation in early learning stage. Based on it, we propose to generate new synthetic samples to robustly train DNNs, by aggregating the original samples with their top-K nearest neighbours.
- We estimate the dynamic weights by unsupervised learning in the creation of synthetic samples. The weights are proportional to the clean probability of samples, thereby maintaining correct information while eliminating the wrong information from the mislabeled samples.
- We demonstrate that the proposed method outperforms the state-of-the-art methods on two standard benchmarks with simulated label noise and two real-world noisy datasets. We also provide extensive ablation study and empirical analyses to verify the effectiveness of different components.

6.4 Related work

Different approaches have been proposed to combating noisy labels in classification task, and they can be classified into the following categories:

Robust loss functions. These studies (Ghosh *et al.*, 2017; Zhang and Sabuncu, 2018; Wang *et al.*, 2019; Ma *et al.*, 2020) focus on developing noise-tolerant loss functions. For instance, Ghosh *et al.* (2017) have proven mean absolute error loss is a noise-tolerant loss function. GCE (Zhang and Sabuncu, 2018) applies a Box-Cox transformation to probabilities which behaves like a generalized mixture of MAE and CE.

Loss correction and Label correction. These approaches either iteratively relabel the noisy labels with their own predictions (Tanaka *et al.*, 2018; Wang *et al.*, 2018; Yi and Wu, 2019) or estimate the noise transition matrix (Patrini *et al.*, 2017). For example, Patrini *et al.* (2017) estimate the noise transition matrix and equally treat all samples to correct the loss. Joint-optim (Tanaka *et al.*, 2018) iteratively updates the labels with soft or hard pseudo-labels of current predictions.

Sample selection by Curriculum Learning. These methods (Jiang *et al.*, 2018; Malach and Shalev-Shwartz, 2017; Han *et al.*, 2018; Lu *et al.*, 2022) effectively train DNNs by selecting samples through a meaningful order (i.e. from easy samples to the hard ones). MentorNet (Jiang *et al.*, 2018) pre-trains a mentor network for selecting clean samples to guide the training of the student network. Co-teaching (Han *et al.*, 2018) symmetrically train two networks by selecting small-loss instances in a mini-batch to teach the other.

Semi-supervised learning and meta-learning. These methods either apply semi-supervised learning techniques after explicitly differentiating noisy samples from training data (Li *et al.*, 2020a; Kim *et al.*, 2019; Nguyen *et al.*, 2020) or use meta-learning (Li *et al.*, 2019). For example, Li *et al.* (2020a) divide the training data into clean and noisy ones, then train two networks with a semi-supervised algorithm MixMatch

(Berthelot *et al.*, 2019). SELF (Nguyen *et al.*, 2020) progressively filters out mislabeled samples with a semi-supervised approach.

Regularization. ELR (Liu *et al.*, 2020) prevents memorization of mislabeled samples by using an explicit regularizer. CAR (Lu *et al.*, 2021) proposes a confidence adaptive regularization to prevent memorization of noisy labels by scaling the gradient.

In contrast to the aforementioned literature, our method trains DNNs on noisy labels without: 1) consulting any clean subset; 2) eliminating training samples; 3) applying augmentation techniques from semi-supervised learning; 4) using any prior information. Specifically, we train DNNs with a new synthetic set created by aggregating the original samples with their top-K nearest neighbours, which prevents DNNs from overfitting to noisy labels during training.

6.4.1 Preliminary

Consider C-class classification problem, we have a noisy training set $\hat{D} = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ is an input and $\hat{\boldsymbol{y}}_i \in \{0,1\}^C$ is the one-hot vector corresponding to $\boldsymbol{x}_i$. Note that the ground truth label $\boldsymbol{y}_i$ is unobservable, and the observable noisy label $\hat{\boldsymbol{y}}_i$ is of certain probability to be incorrect. The classification model maps each input $\boldsymbol{x}_i$ to a C-dimensional logits using a deep neural network model $\mathcal{N}_\Theta$ and then feeds the logits into a softmax function to produce $\boldsymbol{p}_i$ of the conditional probability of each class.

$$\boldsymbol{p}_i = \texttt{softmax}(\mathcal{N}_\Theta(\boldsymbol{x}_i)) = \frac{e^{\mathcal{N}_\Theta(\boldsymbol{x}_i)}}{\sum_{c=1}^{C} e^{(\mathcal{N}_\Theta(\boldsymbol{x}_i))_c}}. \tag{6.1}$$

Θ denotes the parameters of the DNNs and $(\mathcal{N}_\Theta(\boldsymbol{x}_i))_c$ denotes the c-th entry of logits $\mathcal{N}_\Theta(\boldsymbol{x}_i)$. Traditionally, the model $\mathcal{N}_\Theta$ is trained via the cross-entropy (CE) loss to

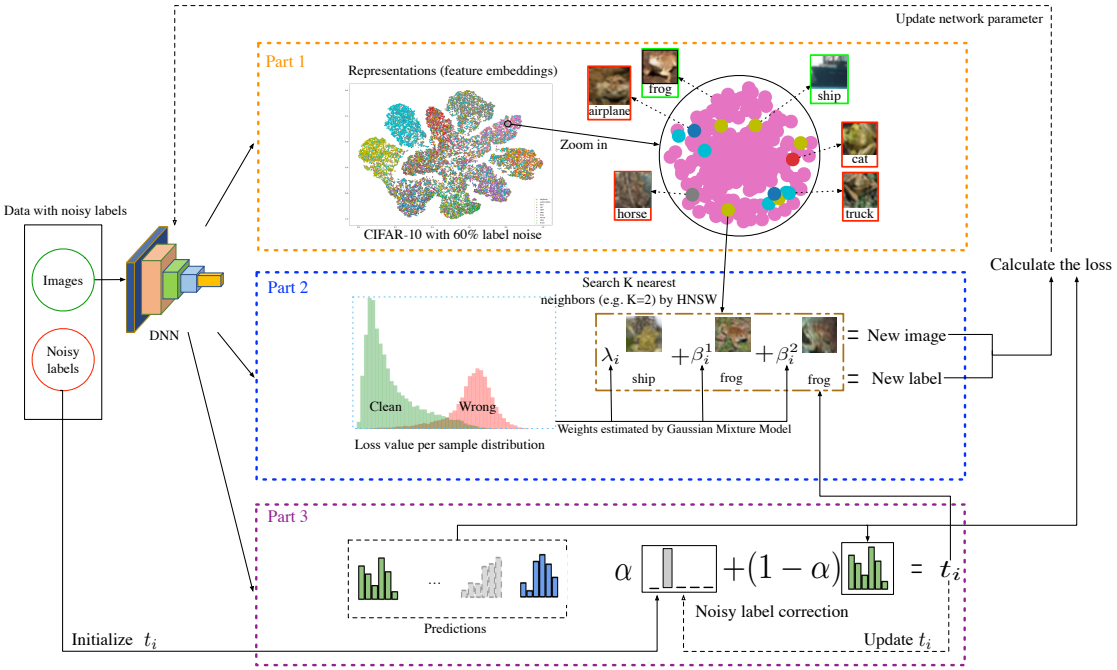


Figure 6.2: The proposed method MixNN consists of three parts. Part 1: Based on the learned representations from the penultimate layer, we calculate each training sample's approximate K-nearest neighbours by using Hierarchical Navigable Small World (HNSW) graph. Part 2: We aggregate the original sample with its K-nearest neighbours by using the dynamic weights estimated from a Gaussian Mixture Model that learned on per-sample loss distribution. Part 3: We gradually correct the noisy labels through an exponential moving average strategy.

measure how well the model fits the training samples.

$$\mathcal{L}_{ce}(\hat{D}, \Theta) = -\frac{1}{N}\sum_{i=1}^{N}\{\ell_{ce}\}^i = -\frac{1}{N}\sum_{i=1}^{N}\hat{\boldsymbol{y}}_i^{\top}\log(\boldsymbol{p}_i). \tag{6.2}$$

6.5 Methodology

We name our framework **MixNN** as it **Mix**es each training sample with its **N**earest **N**eighbours to train the DNNs. A diagram of our framework is shown in Fig. 6.2. Generally, MixNN consists of three parts (boxed with different colors). In this section, we first explore the representation distributions in the early learning stage. Then we describe the details in MixNN, including mixing with nearest neighbours, K-approximate nearest neighbour search, weight estimation, and noisy labels correction.

6.5.1 Representation Distributions

In the noisy label scenario, the training data consist of clean samples and mislabeled samples. The goal of MixNN is to prevent the model from memorizing mislabeled samples while continually learning from clean samples. To study whether the learned representations are corrupted due to label noise, we plot the t-SNE graph (Van der Maaten and Hinton, 2008) of learned representations (i.e. the embeddings from penultimate layer) in the early learning stage in Fig. 6.2 Part 1. As we can see, the learned representations of majority clean samples still congregate in their true classes, while the representations of mislabeled samples disperse in all classes. For clear illustration, we zoom in a random region in frog (pink) class and display the samples in the right as an example. In this region, we observe that most of the frog images have learned correct representations, only a few samples from other classes have learned 'corrupted' representations (e.g. the ship image shouldn't have a fog representation). Besides, most of the mislabeled samples in frog class are also frog images though attached wrong labels from other classes.

This observation motivates us to consider whether we can refer the correct information from the mislabeled samples' nearest neighbours. To achieve our goal, we generate a new training set where each sample is mixed with its K nearest neighours. Ideally, the correct image features are most likely to be preserved while the negative influence of noisy labels can be mitigated, yields achieving robustness to label noise.

6.5.2 Mixing with Nearest Neighbours

Our main idea is to use the correct knowledge from nearest neighbours to mitigate the detrimental impact of noisy labels. For each training sample $\boldsymbol{x}_i$ in a mini-batch, we generate a synthetic training sample $\tilde{\boldsymbol{x}}_i$ by convex linearly combining it with its K nearest neighbours. We denote K nearest neighbours of a training sample $(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$ as $\Phi^K(\boldsymbol{x}_i) = \{(\bar{\boldsymbol{x}}_i^k, \bar{\boldsymbol{y}}_i^k)\}_{k=1}^K = \{(\bar{\boldsymbol{x}}_i^1, \bar{\boldsymbol{y}}_i^1), (\bar{\boldsymbol{x}}_i^2, \bar{\boldsymbol{y}}_i^2), \ldots, (\bar{\boldsymbol{x}}_i^K, \bar{\boldsymbol{y}}_i^K)\}$. Hence, we define the image of a synthetic training sample as

$$\tilde{\boldsymbol{x}}_i = \lambda_i \boldsymbol{x}_i + \sum_{k=1}^{K} \beta_i^k \bar{\boldsymbol{x}}_i^k, \quad \sum_{k=1}^{K} \beta_i^k = 1 - \lambda_i, \tag{6.3}$$

where λ_i is a dynamic scalar value denoting the weight of original training sample. β_i^k denotes the weight of k-th nearest neighbour. We have $\lambda_i + \sum_{k=1}^{K} \beta_i^k = 1$ to ensure the synthetic training sample still follows the same distribution of original sample after normalization. Similarly, we calculate the new label $\tilde{\boldsymbol{y}}_i$ of the new sample by

$$\tilde{\boldsymbol{y}}_i = \lambda_i \hat{\boldsymbol{y}}_i + \sum_{k=1}^{K} \beta_i^k \bar{\boldsymbol{y}}_i^k, \quad \sum_{k=1}^{K} \beta_i^k = 1 - \lambda_i. \tag{6.4}$$

We then train our model with the synthetic training set $\tilde{D} = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)\}_{i=1}^N$. Based on $\tilde{D}$, we use the cross-entropy loss as the measure of how well the model fits the $\tilde{D}$. We

denote

$$\tilde{\boldsymbol{p}}_i = \texttt{softmax}(\mathcal{N}_\Theta(\tilde{\boldsymbol{x}}_i)) = \frac{e^{\mathcal{N}_\Theta(\tilde{\boldsymbol{x}}_i)}}{\sum_{c=1}^{C} e^{(\mathcal{N}_\Theta(\tilde{\boldsymbol{x}}_i))_c}}. \tag{6.5}$$

Thus, the new loss $\mathcal{L}(\tilde{D}, \Theta)$ becomes

$$\begin{aligned} & -\frac{1}{N}\sum_{i=1}^{N} \tilde{\boldsymbol{y}}_i^\top \log(\tilde{\boldsymbol{p}}_i) \\ =& -\frac{1}{N}\sum_{i=1}^{N} (\lambda_i \hat{\boldsymbol{y}}_i + \sum_{k=1}^{K} \beta_i^k \bar{\boldsymbol{y}}_i^k)^T \log(\tilde{\boldsymbol{p}}_i) \\ =& \underbrace{-\frac{1}{N}\sum_{i=1}^{N} \lambda_i \hat{\boldsymbol{y}}_i^\top \log(\tilde{\boldsymbol{p}}_i)}_{\text{first term}} \underbrace{-\frac{1}{N}\sum_{i=1}^{N} (\sum_{k=1}^{K} \beta_i^k \bar{\boldsymbol{y}}_i^k)^\top \log(\tilde{\boldsymbol{p}}_i)}_{\text{second term}}. \end{aligned} \tag{6.6}$$

Compared to CE loss in Eq. (6.2), the new loss is more resistant to noisy labels. In Eq. (6.6), the first term is similar to the CE loss, except it is weighted by λ_i. The target in the second term is mixed by labels from K nearest neighbours. In Section 6.5.4, we will generate $\lambda_i \to 1$ for a clean sample $\boldsymbol{x}_i$, causing $\sum_{k=1}^{K} \beta_i^k \to 0$. Therefore, the new loss reduces to exact CE loss for clean samples. For mislabeled samples, since $\lambda_i \to 0$, the new loss relies on the second term. According to Section 6.5.1, labels of K nearest neighbours $\bar{\boldsymbol{y}}_i^k$ are likely to be the correct labels. The second term penalizes model prediction that are inconsistent with mixed target $\sum_{k=1}^{K} \beta_i^k \bar{\boldsymbol{y}}_i^k$, mitigating the impact of noisy labels on training DNNs.

6.5.3 Approximate Nearest Neighbour Search

In our method, we search the K-Nearest Neighbors (KNN) for each training sample based on the learned representation. Assume the learned representation of a training

sample is a query vector. A naive approach to performing exact KNN search is to directly compute the distances (e.g. Euclidean distance and Cosine distance) between the query and every element in the training set. Hence, the complexity of the naive approach is $O(dN)$, where N is the size of training set and d is the dimension of representation vector.

Previous study (Har-Peled *et al.*, 2012) has demonstrated that exact KNN search solutions may offer a substantial search speedup only in the case of relatively low dimentional data (e.g. $d < 20$) due to "curse of dimensionality". For instance, the complexity of KNN search in KD-tree (Bentley, 1975) is $O(2^d \log(N))$ which is exponential to the dimension of representations. In our case d can be large. For example, the dimension of representation in penultimate layer for ResNet34 (He *et al.*, 2016a) is 512. Therefore, it is inefficient to directly calcuate the exact KNN.

To overcome this problem, a concept of Approximate Nearest Neighbours Search (ANNS) (Har-Peled *et al.*, 2012) was proposed, which relaxes the condition of the exact search by allowing a small number of errors. The quality of an inexact search is defined as the ratio between the number of found true nearest neighbours and K. In this paper, we adopt the Hierarchical Navigable Small World (HNSW) graph (Malkov and Yashunin, 2018) as our search index. It is a fully graph based incremental ANNS structure that can offer a superior logarithmic complexity scaling. The search index in HNSW is a multi-layered structure where each layer is a proximity graph. Each node in the graph corresponds to one of the representations. A nearest neighbour search in HNSW adopts a "zooming-in" style. It starts at an entry point node in the uppermost layer and recursively performs a greedy graph traversal in each layer until it reaches a local minimum in the bottommost one. The maximum number

of connections per element in all layers can be made a constant, thus allowing a logarithmic complexity scaling of routing in a navigable small world graph. In this paper, we use Euclidean distance as the measure of similarity and the overall search complexity scaling is $O(\log(N))$. When $K = 4$, the original exact KNN search costs 5.25 millisecond per sample in our experiment. When using HWSW, the search time is reduced to 0.034 millisecond per sample, which is quite efficient.

6.5.4 Weight Estimation

In mixing functions Eq. (6.3) and Eq. (6.4), the weight λ_i indicates how confidently we can trust the original sample, whereas $\beta_i^1, ..., \beta_i^K$ indicates how much knowledge is referred from these nearest neighbours. Ideally, we want to preserve correct information from the clean samples while dampening the wrong information from the mislabeled samples. In other words, the weights should be able to indicate the 'probability' of a training sample being correctly labeled or not.

Due to the early learning phenomenon, the samples with small-loss values are more likely to be correctly labeled. Therefore, existing sample selection methods (Han *et al.*, 2018) select the clean samples according to the magnitude of loss values. In this paper, we investigate the per-sample loss distribution and find that there is a separation between the loss distribution of correctly labeled samples and loss distribution of mislabeled samples. As shown in Fig. 6.3 (a) and (c), the normalized loss values of the clean samples are in expectation smaller than the mislabeled ones. Even in the case of extreme label noise (e.g. 80% labels are incorrect), their loss distributions can be differentiated. To estimate the probability of a sample being clean, we introduce a two-component Gaussian Mixture Model (GMM) (Permuter

et al., 2006) to fit the per-sample loss distribution as shown in Fig. 6.3 (b) and (d). The probability density function (pdf) of GMM with M components on the per sample loss value ℓ can be defined as

$$P(\ell) = \sum_{m=1}^{M} \pi_m \mathcal{G}(\ell \mid \mu_m, \sigma_m^2), \quad \sum_{m=1}^{M} \pi_m = 1, \tag{6.7}$$

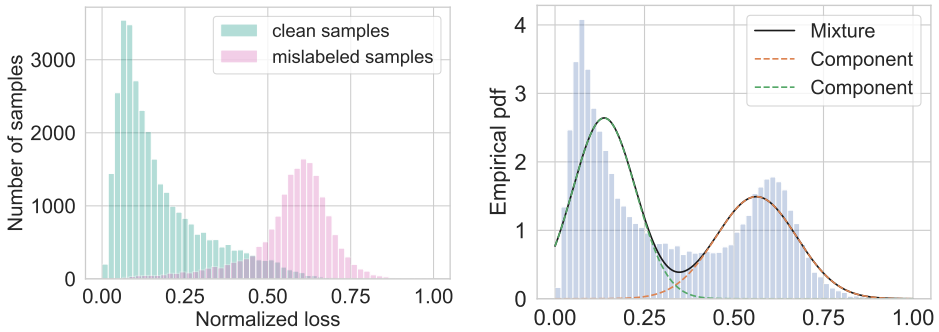
where π_m is the coefficient for the linear convex combination of each individual pdf $\mathcal{G}(\ell \mid \mu_m, \sigma_m^2)$. In our case, we use an Expectation-Maximization (EM) algorithm to estimate the π_m, μ_m and σ_m^2. Therefore, we obtain the probability of a sample being clean or mislabeled through the posterior probability:

$$P(m \mid \ell) = \frac{P(m)P(\ell \mid m)}{P(\ell)} = \frac{\pi_m \mathcal{G}(\ell \mid \mu_m, \sigma_m^2)}{\sum_{m=1}^{M} \pi_m \mathcal{G}(\ell \mid \mu_m, \sigma_m^2)} \tag{6.8}$$

where $m = 0(1)$ indicate correct (wrong) labels. Note that we always calculate the cross-entropy loss to estimate the clean probability for all samples after every epoch. But we use our loss defined in Eq. (6.6) for training the model which contains multiple loss terms to deal with label noise.

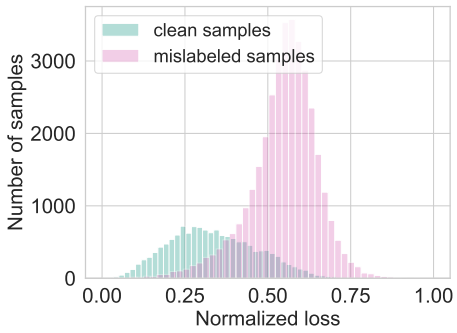
While mislabeled samples benefit from combining with clean ones, clean samples are contaminated by mislabeled ones, whose training objective is incorrectly modified. The goal of mixing strategy in Eq. (6.3) and Eq. (6.4) is to use the dynamic weights to reduce the contribution of mislabeled samples when they are combined with correctly labeled ones. We denote the per sample loss value of $\boldsymbol{x}_i$ as $\ell(\boldsymbol{x}_i)$. Thus the dynamic weights are calculated by

$$\lambda_i = \frac{P(m = 0 \mid \ell(\boldsymbol{x}_i))}{P(m = 0 \mid \ell(\boldsymbol{x}_i)) + \sum_{k=1}^{K} P(m = 0 \mid \ell(\bar{\boldsymbol{x}}_i^k))}, \tag{6.9}$$

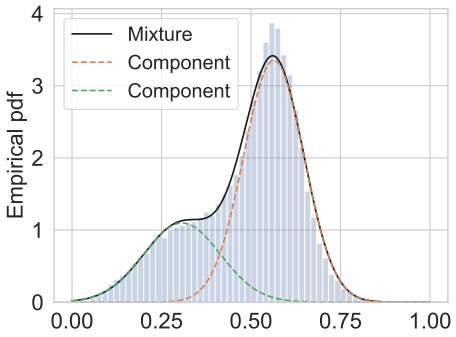


(a) 40% symmetric noise

(b) 40% symmetric noise



(c) 80% symmetric noise



(d) 80% symmetric noise

Figure 6.3: Train on CIFAR-10 with 40% and 80% label noise after 10 epochs with cross-entropy loss. Plots (a) and (c): The ground truth normalized loss distribution. Plots (b) and (d): The pdf of mixture model and two components after fitting a two-component GMM to per-sample loss distribution.

$$\beta_i^k = \frac{P(m=0 \mid \ell(\bar{\boldsymbol{x}}_i^k))}{P(m=0 \mid \ell(\boldsymbol{x}_i)) + \sum_{k=1}^{K} P(m=0 \mid \ell(\bar{\boldsymbol{x}}_i^k))}. \tag{6.10}$$

We then use the above weights to guide the generation of synthetic training sample $(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)$. Consider $K = 1$, there are totally four mixing cases: clean-clean, clean-wrong, wrong-clean, and wrong-wrong. By using dynamic weights, it largely avoids generating the confusing input to the network in clean-wrong and wrong-clean cases, while retaining the strengths for clean-clean and wrong-wrong combinations. More discussion on these four cases is in Section 6.6.

6.5.5 Noisy Labels Correction

Despite that the synthetic training samples set $\tilde{D} = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)\}_{i=1}^N$ is better than directly using given noisy training dataset $\hat{D} = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^N$. Nevertheless, using the noisy labels $\hat{\boldsymbol{y}}_i$ in Eq. (6.4) and Eq. (6.6) may be less effective as $\hat{\boldsymbol{y}}_i$ is likely to be incorrect, especially when the noise rate is extremely high. Therefore, a better estimation of ground truth label $\boldsymbol{y}_i$ can further improve the performance. It has been observe that most predictions in early learning stage are correct. Based on this observation, we use an exponential moving average strategy to gradually estimate the soft target $\boldsymbol{t}_i$ by using the noisy label $\hat{\boldsymbol{y}}_i$ and model prediction $\boldsymbol{p}_i$. We update $\boldsymbol{t}_i$ in each epoch E by

$$\boldsymbol{t}_i = \begin{cases} \hat{\boldsymbol{y}}_i & \text{if } E < E_s \\ \alpha \boldsymbol{t}_i + (1-\alpha)\boldsymbol{p}_i & \text{if } E \geq E_s \end{cases} \tag{6.11}$$

where E_s is the epoch that starts performing label correction and $0 \leq \alpha < 1$ is the momentum. We then replace the noisy label $\hat{\boldsymbol{y}}_i$ in Eq. (6.4) and Eq. (6.6) with the

estimated soft target $\boldsymbol{t}_i$. Consequently, using $\boldsymbol{t}_i$ facilitates the model to memorize more correctly labeled samples and to generate a better new mixed label $\tilde{\boldsymbol{y}}$.

6.6 Case Discussion

To explain how MixNN works, we discuss the possible cases when mixing with one nearest neighbour ($K = 1$). We also provide a toy example in Fig. 6.4 to better illustrate each case. The synthetic samples $(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)$ are generated by the following four ways.

Case 1: *Both* $(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$ *and its nearest neighbour* $(\bar{\boldsymbol{x}}_i^1, \bar{\boldsymbol{y}}_i{}^1)$ *are clean samples.* Based on the analysis of representation distributions in the early learning phase, both $\boldsymbol{x}_i$ and $\bar{\boldsymbol{x}}_i^1$ are most likely to be the images from the same class. In this case, the new mixed input is a convex linear combination of two similar images, i.e., $\tilde{\boldsymbol{x}}_i = \lambda_i \boldsymbol{x}_i + \beta_i^1 \bar{\boldsymbol{x}}_i^1$. And its corresponding label is $\tilde{\boldsymbol{y}}_i = \lambda_i \hat{\boldsymbol{y}}_i + \beta_i^1 \bar{\boldsymbol{y}}_i^1$. Since $\hat{\boldsymbol{y}}_i = \bar{\boldsymbol{y}}_i^1$ and $\lambda_i + \beta_i^1 = 1$, then the mixed label $\tilde{\boldsymbol{y}}_i = \hat{\boldsymbol{y}}_i = \bar{\boldsymbol{y}}_i^1$. Therefore, the new sample $(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)$ is similar to the sample after applying augmentation strategy (Zhang *et al.*, 2018b) which encourages the model to behave linearly in-between training samples, resulting in reducing the number of undesirable oscillations when predicting hard samples. The DNNs trained with such samples will be better calibrated (Thulasidasan *et al.*, 2019b). In other words, the prediction softmax scores are much better indicators of the actual likelihood of predictions, which avoids producing the overconfident wrong predictions and improves the estimation of dynamic weights in Section 6.5.4.

Case 2: $(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$ *is clean sample and its nearest neighbour* $(\bar{\boldsymbol{x}}_i^1, \bar{\boldsymbol{y}}_i{}^1)$ *is mislabeled sample.* Similarly, the new mixed input is most likely to be a combination of two images from the same class, while their labels are inconsistent. In Fig. 6.4 case 2, the

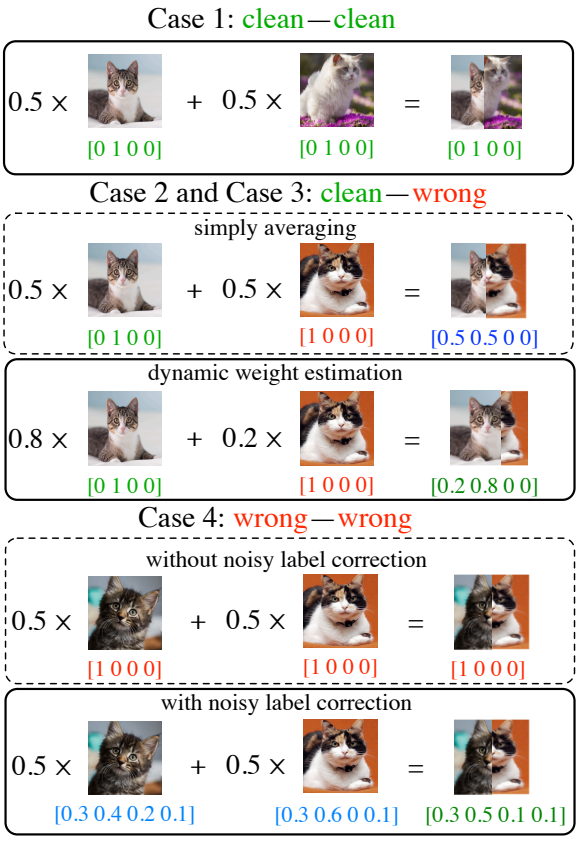


Figure 6.4: A toy example for illustrating the different cases in MixNN. We use the one-hot label vector and each entry indicates four different classes (i.e. dog, cat, fox and monkey).

mixed label becomes an ambiguous target if simply averaging the samples. However, with estimated dynamic weights, the mixed label is determined and correct. Our method to generate the mixed label is similar to label smoothing Szegedy *et al.* (2016) which scales and translates the original noisy label $\hat{\boldsymbol{y}}_i$ to $(1-\gamma)\hat{\boldsymbol{y}}_i + \gamma/C$, but preserves the label with maximal probability when $\gamma < 1$. Different from the label smoothing that uses the fixed uniform distribution to scale the noisy labels, our approach adopts the dynamic weights, where $\lambda, \beta^1, \ldots, \beta^K$ are learned from data, to

adaptively adjust the mixed label for better performance.

Case 3: $(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$ *is mislabeled sample and its nearest neighbour* $(\bar{\boldsymbol{x}}_i^1, \bar{\boldsymbol{y}}_i^1)$ *is clean sample.* This case is similar to case 2, so we do not further discuss it.

Case 4: *Both* $(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$ *and its nearest neighbour* $(\bar{\boldsymbol{x}}_i^1, \bar{\boldsymbol{y}}_i^1)$ *are mislabeled samples.* In this case, the mixed input is combined with two images from the same class, but their labels are both incorrect. Thus the mixed sample is not promised to improve the generalization capacity as shown in Fig. 6.4. However, the noisy labels are very likely to be corrected after using label correction. Therefore, the mixed label can be a reasonable target compared to the original noisy one.

6.7 Experiments

In this section, we first test the efficacy of the proposed method on two benchmark datasets with simulated label noise. Then we provide ablation study and qualitative analyses to investigate the effect of different components. All experiments are implemented in Pytorch and run on a single NVIDIA A100 GPU.

6.7.1 Effectiveness on Simulated Label Noise

We conduct the experiments with simulated label noise on the two benchmarks *CIFAR-10* and *CIFAR-100*.

Label Noise Simulation. Given CIFAR-10 and CIFAR-100 are initially clean, we follow the way in (Patrini *et al.*, 2017) to corrupt these two benchmarks manually by label transition matrix $\mathbf{Q}$, where $\mathbf{Q}_{ij} = \Pr[\hat{y} = j \mid y = i]$ denotes the probability that noisy label $\hat{y}$ is flipped from clean label y. Generally, the matrix $\mathbf{Q}$ has two

representative label noise models. (1) Symmetric noise is generated by uniformly flipping labels in each class to one of the other class labels with probability ε. (2) Asymmetric noise is a simulation of fine-grained classification with noisy labels in the real world, where the annotators are more likely to make mistakes only within very similar classes. In this paper, the asymmetric noisy labels are generated by flipping *truck* $\rightarrow$ *automobile*, *bird* $\rightarrow$ *airplane*, *deer* $\rightarrow$ *horse* and *cat* $\leftrightarrow$ *dog* for CIFAR-10. For CIFAR-100, the noise flips each class into the next, circularly within super-classes.

Network and Optimizer. We use ResNet34 (He *et al.*, 2016a) for both datasets, and train them using SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 128. The networks are trained for 300 epochs. We use the cosine annealing learning rate (Loshchilov and Hutter, 2016) where the maximum number of epoch for each period is 10, the maximum and minimum learning rate is set to 0.02 and 0.001 respectively. We warm up our networks 10 epochs for CIFAR-10 and 30 epochs for CIFAR-100 with cross-entropy loss. We do not perform early stopping since we don't assume the presence of clean validation data. All test accuracies are recorded from the last epoch of training. The reason that we train the model 300 epochs is to fully evaluate whether the model will memorize the mislabeled samples, avoiding the interference caused by early stopping (Li *et al.*, 2020b).

Hyperparameters. We set $K = 1$ since we find that a larger K worsens the performance. E_s should be smaller than the epoch that model starts memorizing mislabeled samples. We estimate this epoch by (Lu and He, 2022). In this paper, we fix $E_s = 60$ and $\alpha = 0.9$ by default. More discussions on choice of K and α can be found in Section 6.7.4.

Results on CIFAR-10 and CIFAR-100 Table 6.1 shows the classification test

Dataset	CIFAR-10					CIFAR-100				
Noise type	symm				asymm	symm				asymm
Method/Noise ratio	20%	40%	60%	80%	40%	20%	40%	60%	80%	40%
CE	86.98 ± 0.12	81.88 ± 0.29	74.14 ± 0.56	53.82 ± 1.04	80.11 ± 1.44	58.72 ± 0.26	48.20 ± 0.65	37.41 ± 0.94	18.10 ± 0.82	42.74 ± 0.61
F-correction (Patrini *et al.*, 2017)	87.99 ± 0.36	83.25 ± 0.38	74.96 ± 0.65	54.64 ± 0.44	83.55 ± 0.58	39.19 ± 2.61	31.05 ± 1.44	19.12 ± 1.95	8.99 ± 0.58	34.44 ± 1.93
Bootstrap (Reed *et al.*, 2015)	86.23 ± 0.23	82.23 ± 0.37	75.12 ± 0.56	54.12 ± 1.32	81.21 ± 1.47	58.27 ± 0.21	47.66 ± 0.55	34.68 ± 1.10	21.64 ± 0.97	45.12 ± 0.57
GCE (Zhang and Sabuncu, 2018)	89.83 ± 0.20	87.13 ± 0.22	82.54 ± 0.23	64.07 ± 1.38	76.74 ± 0.61	66.81 ± 0.42	61.77 ± 0.24	53.16 ± 0.78	29.16 ± 0.74	47.22 ± 1.15
SCE (Wang *et al.*, 2019)	89.83 ± 0.32	87.13 ± 0.26	82.81 ± 0.61	68.12 ± 0.81	82.51 ± 0.45	70.38 ± 0.13	62.27 ± 0.22	54.82 ± 0.57	25.91 ± 0.44	49.32 ± 0.87
NCE+RCE (Ma *et al.*, 2020)	-	86.02 ± 0.09	79.78 ± 0.50	52.71 ± 1.90	79.59 ± 0.40	-	59.48 ± 0.56	47.12 ± 0.62	25.80 ± 1.12	46.79 ± 0.96
Mixup (Zhang *et al.*, 2018b)	93.58	89.46	78.32	66.32	81.66	69.31	58.12	41.10	18.77	49.61
Joint Optim (Tanaka *et al.*, 2018)	92.25	90.79	86.87	69.16	-	58.15	54.81	47.94	17.18	-
PENCIL (Yi and Wu, 2019)	-	-	-	-	**91.01**	-	69.12 ± 0.62	57.79 ± 3.86	fail	63.61 ± 0.23
RoG+D2L (Lee *et al.*, 2019)	-	87.00	78.00	-	-	-	64.90	40.60	-	-
M-correction (Arazo *et al.*, 2019)	-	92.30	86.10	74.10	-	-	70.10	59.50	39.50	-
MentorNet (Jiang *et al.*, 2018)	92.00	91.20	74.20	60.00	-	73.50	68.50	61.20	35.50	-
O2U-net (Huang *et al.*, 2019)	-	90.30	-	43.40	-	-	69.20	-	39.40	-
NLNL (Kim *et al.*, 2019)	**94.23**	92.43	88.32	-	89.86	71.52	66.39	56.51	-	45.70
DAC (Thulasidasan *et al.*, 2019a)	92.91	90.71	86.30	74.84	-	73.55	66.92	57.17	32.16	-
SELF (Nguyen *et al.*, 2020)	-	91.13	-	63.59	-	-	66.71	-	35.56	-
ELR (Liu *et al.*, 2020)	91.16 ± 0.08	89.15 ± 0.17	86.12 ± 0.49	73.86 ± 0.61	90.12 ± 0.47	74.21 ± 0.22	68.28 ± 0.31	59.28 ± 0.67	29.78 ± 0.56	**73.26 ± 0.64**
MixNN (Ours)	93.91 ± 0.12	**92.89 ± 0.02**	**91.66 ± 0.07**	**86.08 ± 1.01**	90.25 ± 0.76	**74.81 ± 0.14**	**72.97 ± 0.14**	**67.56 ± 0.17**	**48.81 ± 0.06**	68.18 ± 0.11

Table 6.1: Test Accuracy (%) on CIFAR-10 and CIFAR-100 with different ratios of symmetric and asymmetric label noise. We compare with existing methods under the same backbone ResNet34. The average accuracy and standard deviation of 3 random runs are reported. symm/asymm represent symmetric/asymmetric label noise respectively. **Bold** indicates the best results. - indicates the result is not reported.

accuracies of our approach on CIFAR-10 and CIFAR-100 with different levels of symmetric and asymmetric label noise. As we can see, MixNN achieves the highest accuracy in most cases, especially in challenging ones. For example, on CIFAR-10 with 80% symmetric label noise, MixNN outperforms the best state-of-the-art method (74.84% of DAC) by more than 11%. In the hardest case (i.e. CIFAR-100 with 80% symmetric label noise), we observe that most existing methods achieve relatively low test accuracies and PENCIL even fails to converge. However, MixNN still achieves the best accuracy up to 48.81%. Note that on CIFAR-10/CIFAR-100 with 20% symmetric label noise, NLNL and DAC obtain superior performance and even outperform our approach. However, these two methods are more complex and perform multiple training stages for different purposes. For example, NLNL performs three training stages including a) Division of training data into either clean or noisy data with a DNN model. b) Training initialized DNN with clean data from the first stage and then updating noisy data's label following the output of DNN trained with clean

data. c) Clean data and label-updated noisy data are both used for training initialized DNN in the final stage. In contrast, our method conducts an end-to-end learning manner which is much simpler than NLNL and also achieves excellent performance. In summary, MixNN shows a consistently strong performance across all datasets with different types and ratios of simulated label noise.

6.7.2 Ablation Study

We study the effect of removing different components to provide insights into what makes MixNN successful. The results are in Table 6.2. First, we remove the dynamic weight estimation in MixNN. Instead, we average (AVG) the selected samples or use random (RDM) weights from a Beta distribution in mixing function. We observe that merely averaging the selected samples does not perform well in all noisy cases, especially when the noise ratio is large. In comparison, using random weights may result in a surprising performance. For instance, it achieves excellent performance in CIFAR-100 with 40% asymmetric label noise. Since asymmetric label noise is concentrated inside each class, the estimated weights may fail to capture the clean probability of difficult samples, forcing the mixing function to use an average way. However, using the random weights is likely to appropriately assign the weights for the challenging samples, reducing the impact of asymmetric label noise. We also remove the label correction to see how it affects the performance. Without label correction, the performance suffers, especially when the noise ratio is high. Further, we investigate the performance of MixNN when using random samples instead of using K nearest neighbours. We observe that the performance is marginally poorer in the low ratio of label noise. However, when the training data contains high ratios

Dataset	CIFAR-10			CIFAR-100		
Noise type	symm		asymm	symm		asymm
Noise ratio	40%	80%	40%	40%	80%	40%
Baseline CE	81.88 ± 0.29	53.82 ± 1.04	80.11 ± 1.44	48.20 ± 0.65	18.10 ± 0.82	42.74 ± 0.61
Mixup	89.46 ± 0.47	66.32 ± 0.92	81.66 ± 1.73	58.12 ± 0.87	18.77 ± 1.14	49.61 ± 1.39
MixNN	**92.89 ± 0.02**	**86.08 ± 1.01**	**90.25 ± 0.76**	**72.97 ± 0.14**	**48.81 ± 0.06**	68.18 ± 0.11
MixNN w/o weight estimation (AVG)	88.79 ± 0.33	58.53 ± 0.15	82.27 ± 0.10	64.21 ± 0.19	27.91 ± 0.31	59.73 ± 0.21
MixNN w/o weight estimation (RDM)	91.92 ± 0.09	78.72 ± 0.13	89.91 ± 0.27	71.08 ± 0.25	41.14 ± 0.18	**72.98 ± 0.47**
MixNN w/o label correction	91.73 ± 0.01	74.31 ± 0.49	88.73 ± 0.11	66.09 ± 0.50	30.18 ± 0.52	63.66 ± 0.43
MixNN w/o K nearest neighbours	92.19 ± 0.04	76.58 ± 0.25	88.69 ± 0.17	71.46 ± 0.08	42.02 ± 0.48	68.61 ± 0.16

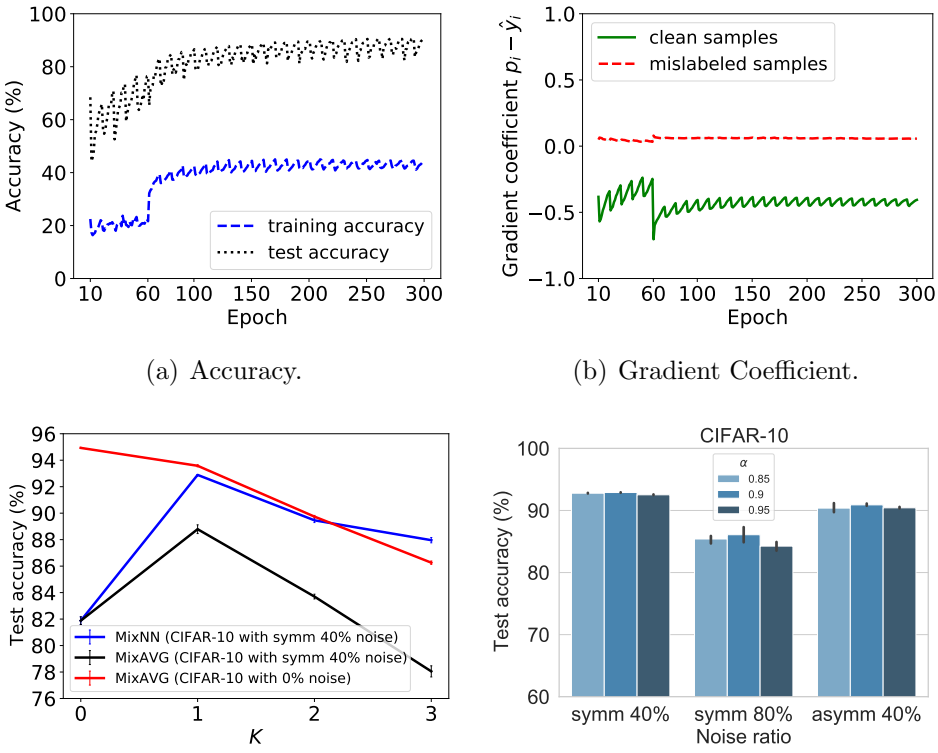
Table 6.2: Ablation study results in terms of test accuracy (%) on CIFAR-10 and CIFAR-100.

of noisy labels, the performance suffers a large decline, demonstrating the benefit of using K nearest neighbours in MixNN.

6.7.3 Learning Stability and Gradient Analysis

In section 3.5, we have demonstrated the failure of CE when trained DNNs with noisy labels. The training accuracy constantly increases indicates the DNNs eventually overfit the noisy labels, resulting in a drop of accuracy on the clean test set. To verify the denoising effect of MixNN, we show its learning stability by plotting the training and test accuracy vs. the number of epochs in Fig. 6.5(a) on the CIFAR-10 with 60% label noise. We observe that the training accuracy stabilizes at around 40% after 60 epochs. It means the DNNs only fit the clean samples, resulting in no drop in test accuracy.

We further investigate the gradient coefficient of clean and mislabeled samples when using MixNN. In Fig. 6.5(b), we observe that the gradient of mislabeled samples is close to 0, letting the gradient of clean samples dominates the whole gradient throughout the training. Therefore, MixNN forces the model to learn from clean samples rather than mislabeled samples. In addition, we notice that there is an



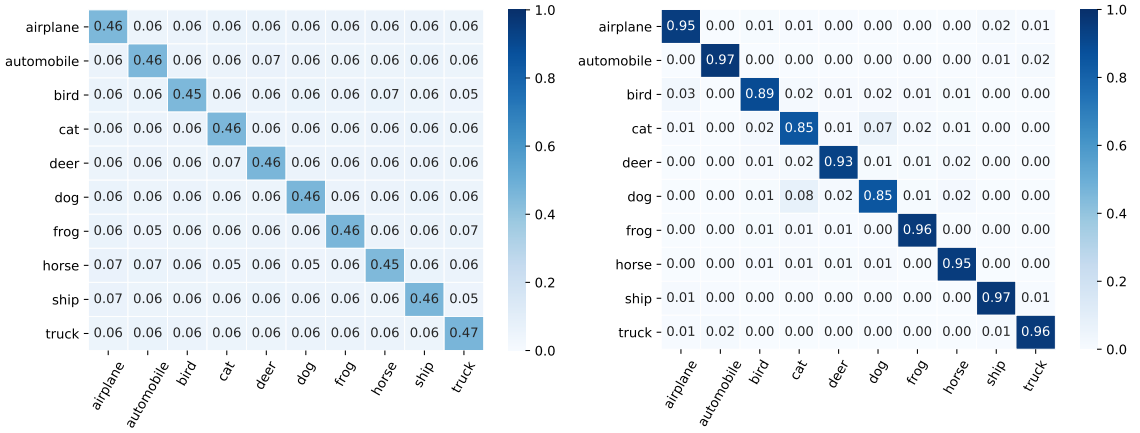
(a) Accuracy.

(b) Gradient Coefficient.

(c) The influence of K.

(d) Sensitivity of α.

Figure 6.5: For plots (a) and (b), we train MixNN on CIFAR-10 with 60% symmetric label noise. Plot (a) shows the training and test accuracy vs. the number of epochs. Plot (b) shows the gradient coefficient vs. the number of epochs. Plot (c) shows the effect of different K on performance. Plot (d) shows the sensitivity of α on performance.



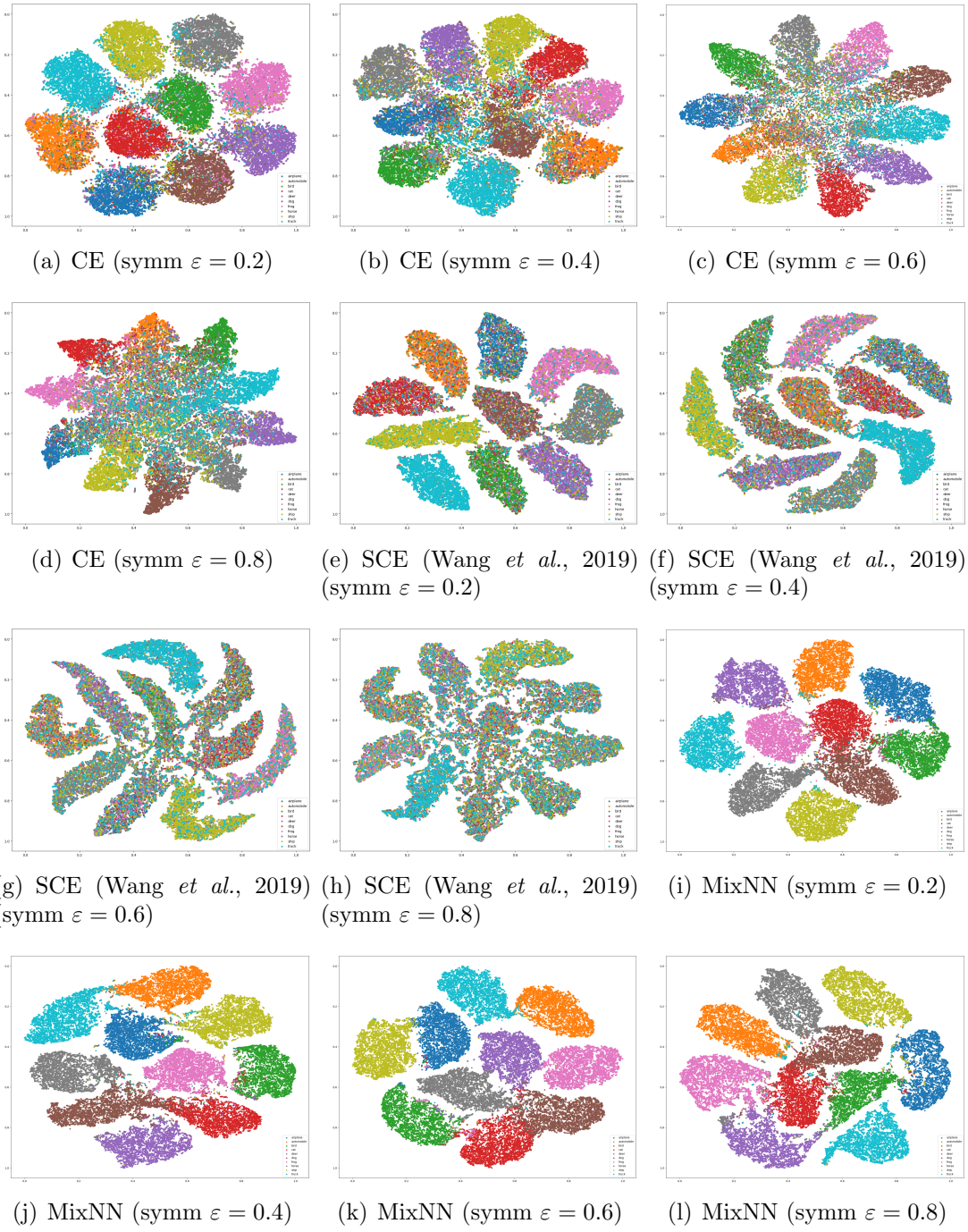
(a) Original (b) After MixNN

Figure 6.6: Confusion matrix of corrected labels w.r.t clean labels on CIFAR-10 with 60% symmetric label noise. The y-axis is the clean labels. The x-axis is the original noisy labels and corrected labels for plots (a) and (b), respectively.

obvious gradient drop for clean samples when starting using noisy label correction (Note that we start label correction from epoch 60), which demonstrates the label correction effectively suppresses gradient inversion.

6.7.4 Choice of K and α

The mixing functions Eq. 6.3 and Eq. 6.4 contain a parameter K to control the number of nearest neighbours mixed with original sample. To study the effect of K on our approach, we test different K on CIFAR-10. As shown in Fig. 6.5(c), MixNN achieves the worst performance as it reduces to regular training (i.e. CE) when $K = 0$. However, the performance of MixNN starts decreasing with the continuing increase of K. To explore whether the performance drop is due to dynamic weight estimation, we add another set of experiments where we simply use averaging strategy (abbreviated as MixAVG). We observe that MixAVG has a similar tendency on performance with



(a) CE (symm $\varepsilon = 0.2$) (b) CE (symm $\varepsilon = 0.4$) (c) CE (symm $\varepsilon = 0.6$)

(d) CE (symm $\varepsilon = 0.8$) (e) SCE (Wang *et al.*, 2019) (symm $\varepsilon = 0.2$) (f) SCE (Wang *et al.*, 2019) (symm $\varepsilon = 0.4$)

(g) SCE (Wang *et al.*, 2019) (symm $\varepsilon = 0.6$) (h) SCE (Wang *et al.*, 2019) (symm $\varepsilon = 0.8$) (i) MixNN (symm $\varepsilon = 0.2$)

(j) MixNN (symm $\varepsilon = 0.4$) (k) MixNN (symm $\varepsilon = 0.6$) (l) MixNN (symm $\varepsilon = 0.8$)

Figure 6.7: t-SNE plots of feature representations learned by CE, SCE, and our proposed method MixNN on CIFAR-10 with different ratios of label noise. Different colours represent the different classes in CIFAR-10.

the increase of K. Therefore, dynamic weight estimation is not the reason causing performance drop. We then conjecture that the reason is the over-mixture of input images. Since the resulting image is the mixture of K images, larger K leads to a more complex input. To avoid the influence of noisy labels, we conduct another set of experiments that use MixAVG on clean training data. We observe a similar performance drop when increasing K. Therefore, a reasonable K can suit our needs, while a large K makes it more difficult for the model to learn from complicated inputs, resulting in performance degradation. Figure 6.5(d) shows the test accuracy when using different $\alpha \in \{0.85, 0.9, 0.95\}$ on CIFAR-10. We observe that the sensitivity to hyperparameter α is quite mild.

6.7.5 Effectiveness of Noisy Label Correction

In Section 6.5.5, the estimated target $\boldsymbol{t}_i$ is calculated by an exponential moving average between the given noisy labels and model predictions, thus our method can gradually correct the noisy labels. We use the confusion matrix to evaluate the quality of labels before and after using our method. In Figure 6.6 shows the results on CIFAR-10 with 60% symmetric label noise. Our approach corrects the noisy labels impressively well for all classes. Furthermore, our mixing strategy is more reliable to obtain the accurate mixed labels using the corrected soft labels. We also calculate the correction accuracy of our strategy. Correction accuracy can be calculated by $\frac{1}{N}\sum_i^N \mathbb{1}\{\text{argmax } \boldsymbol{y}_i = \text{argmax } \boldsymbol{t}_i\}$, where $\boldsymbol{y}_i$ is the ground truth label of training sample $\boldsymbol{x}_i$. We evaluate the correction accuracy on CIFAR-10 and CIFAR-100 with 40% symmetric label noise. Our approach successfully corrects a huge amount of labels and obtains recovered accuracy of 95.25% and 84.86%, respectively. More results on other noise ratios can

be found in supplementary materials.

6.7.6 Feature Representations

We further investigate the representations learned by our approach compared to that learned by CE loss and SCE (Wang *et al.*, 2019). We extract the high-dimensional representations at the penultimate layer and project them to a 2D embedding by using t-SNE (Van der Maaten and Hinton, 2008). The projected representations are illustrated in Fig. 6.7 for 20%, 40%, 60% and 80% symmetric label noise respectively. Under all settings, the feature representations learned by our approach are of significantly better quality than that of CE and SCE with more separated and clearly bounded clusters. We find that SCE always keeps the feature representations of mislabeled samples in their true classes, the same as what CE does in the early learning stage, which prevents the model from memorizing them. However, the boundary formed by the SCE becomes increasingly blurred as the noise ratio rises (see $\varepsilon = 0.8$ case). In contrast, our approach gradually corrects the noisy labels to clean labels, resulting in most of the feature representations in different classes are corrected.

6.8 Conclusion

In this paper, we explore the representation distribution in the early learning stage and propose a method called MixNN for robust learning with noisy labels. MixNN mitigates the negative influence of noisy labels by training with the synthetic samples obtained by mixing the original training samples with their K nearest neighbours.

There are two key elements to our approach. The mixing strategy is dynamically adjusted by the learned Gaussian Mixture Model on per-sample loss distribution. The soft targets in our loss function are gradually corrected by using an exponential moving average on the given labels and model predictions. Through extensive experiments across multiple datasets with simulated and real-world label noise, we demonstrate that MixNN consistently exhibits substantial performance improvements compared to state-of-the-art methods. Importantly, the proposed approach works with any classifier "out-of-the-box" without any changes to architecture or training procedure. In the future, we are interested in adapting MixNN to other research domains such as natural language process (NLP) and object detection, and believe MixNN is a promising algorithm for training robust DNNs against label noise.

Chapter 7

Conclusion and Future Work

The core issue for solving learning with noisy labels is reducing the learning error. Through the error analysis, we decompose the learning error into approximation error, estimation error and fitting error. In this thesis, we propose five different approaches to reduce estimation error and fitting error effectively.

We first propose Co-matching, which uses two networks with different strengths of augmented inputs to keep divergence and to achieve better ensemble effect. To avoid the influence of noisy labels, we introduce an unsupervised matching loss for knowledge distillation. In addition, we fit a mixture model to sample loss distribution to select the clean samples without the need of known noise rates. We provided the theoretical analysis on our loss functions and demonstrate the effectiveness of Co-matching on both benchmark and real-world datasets.

Although Co-matching achieves impressive performance, it requires more computation power (two networks) and the noisy labels are not corrected. We then propose a simple and effective method SELC. SELC leverages the model predictions of previous epochs to correct the noisy labels, thus preventing the model from overfitting

to noisy labels. By evaluating SELC on different types of label noise, we observe its superior performance over existing approaches. Also, the labels after using SELC are much cleaner than the original noisy labels.

Since SELC utilizes the ensemble predictions, it requires an extra space to store them. We then propose a regularization method NAL. NAL introduces an attention branch and a noise attention loss to learn the attention weights for distinguishing the mislabeled samples from clean samples. NAL can effectively diminish the gradient of mislabeled samples, mitigating the effect of noisy labels. We also provide extensive empirical analyses and evaluate its effectiveness across multiple datasets with different types and ratios of label noises.

Since NAL requires a extra branch to learn the attention weights, we then propose a simpler approach: EM-enhanced loss functions. We observe the fluctuation of model output in learning with noisy labels and propose to stabilize the model output for consistent predictions during training. We revealed a new insight into noise-robust loss functions that: EM can make non-robust loss functions robust to label noise without suffering from the underfitting problem, outperforming most existing robust loss functions such as MAE, RCE and NCE. We provide both empirical and theoretical analyses to verify the effectiveness of proposed method. Moreover, our experimental results demonstrate that the proposed method achieves superior performance over the state-of-the-art losses on both simulated and real-world datasets.

Finally, we explore the representation distribution in the early learning stage and propose a method called MixNN for robust learning with noisy labels. MixNN mitigates the negative influence of noisy labels by training with the synthetic samples obtained by mixing the original training samples with their K nearest neighbors.

Our methods	Universal	Computation resource	Training complexity	Performance	#stars	Require noise assumption	Theoretical guarantee
Co-matching (Lu *et al.*, 2022)	⋆⋆	⋆⋆	⋆⋆	⋆⋆	8	No	Partial
SELC (Lu and He, 2022)	⋆⋆⋆	⋆⋆	⋆⋆⋆	⋆⋆⋆	11	No	No
NAL (Lu *et al.*, 2021)	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆	11	No	Partial
EM-enhanced loss	⋆	⋆⋆⋆	⋆⋆⋆	⋆	8	Yes	Fully
MixNN (Lu and He, 2021)	⋆⋆	⋆⋆	⋆	⋆⋆⋆	8	No	No

Table 7.1: Comparison of the proposed methods. ⋆, ⋆⋆, and ⋆ ⋆ ⋆ stand for Bad, Medium, and Good.

There are two key elements to our approach. The mixing strategy is dynamically adjusted by the learned Gaussian Mixture Model on per-sample loss distribution. The soft targets in our loss function are gradually corrected by using an exponential moving average on the given labels and model predictions. Through extensive experiments across multiple datasets with simulated and real-world label noise, we demonstrate that MixNN consistently exhibits substantial performance improvements compared to state-of-the-art methods.

In summary, we compare all proposed methods regarding the universal, computation resource, training complexity, and performance in Table 7.1. There are still multiple open problems for future research. Currently, label noise has been extensively studied in image classification task. Other research areas, such as graph learning and federated learning, remain to be explored. On the methodological front, we hope that our works will trigger interest in designing new techniques, such as network architectures or loss functions, that inherently provide robustness to label noise.

Bibliography

Arazo, E., Ortego, D., Albert, P., O'Connor, N., and Mcguinness, K. (2019). Unsupervised label noise modeling and loss correction. In *Proceedings of the 36th International Conference on Machine Learning*, pages 312–321.

Arpit, D., Jastrzebski, S. K., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A. C., Bengio, Y., *et al.* (2017). A closer look at memorization in deep networks. In *ICML*.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, **18**(9), 509–517.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. (2019). Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060.

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

Bossard, L., Guillaumin, M., and Gool, L. V. (2014). Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pages 446–461. Springer.

Bottou, L. and Bousquet, O. (2007). The tradeoffs of large scale learning. *Advances in neural information processing systems*, **20**.

Chen, P., Liao, B. B., Chen, G., and Zhang, S. (2019). Understanding and utilizing deep neural networks trained with noisy labels. In *International Conference on Machine Learning*, pages 1062–1070.

Chen, P., Ye, J., Chen, G., Zhao, J., and Heng, P.-A. (2021a). Beyond class-conditional assumption: A primary attempt to combat instance-dependent label noise. In *AAAI*, volume 35, pages 11442–11450.

Chen, P., Ye, J., Chen, G., Zhao, J., and Heng, P.-A. (2021b). Robustness of accuracy metric and its inspirations in learning with noisy labels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11451–11461.

Chen, Y., Shen, X., Hu, S. X., and Suykens, J. A. (2021c). Boosting co-teaching with compression regularization for label noise. In *CVPR*, pages 2688–2692.

Cheng, J., Liu, T., Ramamohanarao, K., and Tao, D. (2020). Learning with bounded instance and label-dependent label noise. In *International Conference on Machine Learning*, pages 1789–1799. PMLR.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 113–123.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Ding, Y., Wang, L., Fan, D., and Gong, B. (2018). A semi-supervised two-stage approach to learning from noisy labels. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1215–1224. IEEE.

Englesson, E. and Azizpour, H. (2021). Generalized jensen-shannon divergence loss for learning with noisy labels. *Advances in Neural Information Processing Systems*, **34**.

Feng, L., Shu, S., Lin, Z., Lv, F., Li, L., and An, B. (2021). Can cross entropy loss be robust to label noise? In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 2206–2212.

Frénay, B. and Verleysen, M. (2013). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, **25**(5), 845–869.

Garg, S., Balakrishnan, S., Kolter, Z., and Lipton, Z. (2021). Ratt: Leveraging unlabeled data to guarantee generalization. In *International Conference on Machine Learning*, pages 3598–3609. PMLR.

Ghosh, A., Kumar, H., and Sastry, P. (2017). Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 1919–1925.

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

Grandvalet, Y. and Bengio, Y. (2005). Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536.

Gui, X.-J., Wang, W., and Tian, Z.-H. (2021). Towards understanding deep learning from noisy labels with small-loss criterion. *IJCAI*.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. (2018). Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537.

Har-Peled, S., Indyk, P., and Motwani, R. (2012). Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory of computing*, **8**(1), 321–350.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer.

Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. (2018). Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in neural information processing systems*, pages 10456–10465.

Hu, W., Li, Z., and Yu, D. (2019). Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*.

Huang, J., Qu, L., Jia, R., and Zhao, B. (2019). O2u-net: A simple noisy label detection approach for deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3326–3334.

Huang, L., Zhang, C., and Zhang, H. (2020). Self-adaptive training: beyond empirical risk minimization. *arXiv preprint arXiv:2002.10319*.

Iscen, A., Valmadre, J., Arnab, A., and Schmid, C. (2022). Learning with neighbor consistency for noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4672–4681.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8580–8589.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. (2018). Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313.

Kim, T., Ko, J., Cho, S., Choi, J., and Yun, S.-Y. (2021). Fine samples for learning with noisy labels. *NIPS*.

Kim, Y., Yim, J., Yun, J., and Kim, J. (2019). Nlnl: Negative learning for noisy labels. In *ICCV*, pages 101–110.

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

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Laine, S. and Aila, T. (2016). Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*.

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

LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**(11), 2278–2324.

Lee, K., Yun, S., Lee, K., Lee, H., Li, B., and Shin, J. (2019). Robust inference via generative classifiers for handling noisy labels. In *International Conference on Machine Learning*, pages 3763–3772. PMLR.

Lee, K.-H., He, X., Zhang, L., and Yang, L. (2018). Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5447–5456.

Li, J., Wong, Y., Zhao, Q., and Kankanhalli, M. S. (2019). Learning to learn from noisy labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5051–5059.

Li, J., Socher, R., and Hoi, S. C. (2020a). Dividemix: Learning with noisy labels as semi-supervised learning. *International Conference on Learning Representations*.

Li, M., Soltanolkotabi, M., and Oymak, S. (2020b). Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International conference on artificial intelligence and statistics*, pages 4313–4324. PMLR.

Li, W., Wang, L., Li, W., Agustsson, E., and Van Gool, L. (2017). Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Liu, S., Niles-Weed, J., Razavian, N., and Fernandez-Granda, C. (2020). Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, **33**.

Liu, X., Wang, Z., Shao, J., Wang, X., and Li, H. (2019). Improving referring expression grounding with cross-modal attention-guided erasing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1950–1959.

Liu, Y. (2021). Understanding instance-level label noise: Disparate impacts and treatments. In *International Conference on Machine Learning*, pages 6725–6735. PMLR.

Lloyd, R. V., Erickson, L. A., Casey, M. B., Lam, K. Y., Lohse, C. M., Asa, S. L., Chan, J. K., DeLellis, R. A., Harach, H. R., Kakudo, K., *et al.* (2004). Observer variation in the diagnosis of follicular variant of papillary thyroid carcinoma. *The American journal of surgical pathology*, **28**(10), 1336–1340.

Loshchilov, I. and Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Lu, Y. and He, W. (2021). Mixnn: Combating noisy labels in deep learning by mixing with nearest neighbors. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 847–856. IEEE.

Lu, Y. and He, W. (2022). Selc: Self-ensemble label correction improves learning with noisy labels. *IJCAI*.

Lu, Y., Bo, Y., and He, W. (2021). Confidence adaptive regularization for deep learning with noisy labels. *arXiv preprint arXiv:2108.08212*.

Lu, Y., Bo, Y., and He, W. (2022). An ensemble model for combating label noise. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 608–617.

Lyu, Y. and Tsang, I. W. (2019). Curriculum loss: Robust learning and generalization against label corruption. In *International Conference on Learning Representations*.

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S., Xia, S., Wijewickrema, S., and Bailey, J. (2018). Dimensionality-driven learning with noisy labels. In *International Conference on Machine Learning*, pages 3355–3364.

Ma, X., Huang, H., Wang, Y., Romano, S., Erfani, S., and Bailey, J. (2020). Normalized loss functions for deep learning with noisy labels. In *International Conference on Machine Learning*, pages 6543–6553. PMLR.

Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., and Van Der Maaten, L. (2018). Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196.

Malach, E. and Shalev-Shwartz, S. (2017). Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pages 960–970.

Malkov, Y. A. and Yashunin, D. A. (2018). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, **42**(4), 824–836.

Mirzasoleiman, B., Cao, K., and Leskovec, J. (2020). Coresets for robust training of neural networks against noisy labels. *NIPS*.

Mitchell, T. M. *et al.* (1997). Machine learning.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. (2013). Learning with noisy labels. *NIPS*, **26**, 1196–1204.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. *NeurIPS*.

Nguyen, T., Mummadi, C., Ngo, T., Beggel, L., and Brox, T. (2020). Self: learning to filter noisy labels with self-ensembling. In *International Conference on Learning Representations (ICLR)*.

Nishi, K., Ding, Y., Rich, A., and Hollerer, T. (2021). Augmentation strategies for learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8022–8031.

Northcutt, C. G., Athalye, A., and Mueller, J. (2021). Pervasive label errors in test sets destabilize machine learning benchmarks. *NIPS*.

Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. (2017). Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., and Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. *ICLR*.

Permuter, H., Francos, J., and Jermyn, I. (2006). A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern recognition*, **39**(4), 695–706.

Pleiss, G., Zhang, T., Elenberg, E., and Weinberger, K. Q. (2020). Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, **33**, 17044–17056.

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

Ren, M., Zeng, W., Yang, B., and Urtasun, R. (2018). Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343.

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., and Aroyo, L. M. (2021). "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15.

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

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *ICLR*.

Song, H., Kim, M., and Lee, J.-G. (2019). Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, pages 5907–5915.

Song, J., Dauphin, Y., Auli, M., and Ma, T. (2020). Robust and on-the-fly dataset denoising for image classification. In *European Conference on Computer Vision*, pages 556–572. Springer.

Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., and Fergus, R. (2015). Training convolutional networks with noisy labels. *ICLR*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. (2018). Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5552–5560.

Tanno, R., Saeedi, A., Sankaranarayanan, S., Alexander, D. C., and Silberman, N. (2019). Learning from noisy labels by regularized estimation of annotator confusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11244–11253.

Tarvainen, A. and Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1195–1204.

Thulasidasan, S., Bhattacharya, T., Bilmes, J., Chennupati, G., and Mohd-Yusof, J. (2019a). Combating label noise in deep learning using abstention. In *ICML*, pages 6234–6243. PMLR.

Thulasidasan, S., Chennupati, G., Bilmes, J. A., Bhattacharya, T., and Michalak, S. (2019b). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Advances in Neural Information Processing Systems*, **32**.

Turing, A. M. (2009). Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer.

Vahdat, A. (2017). Toward robustness against label noise in training deep discriminative neural networks. In *Advances in Neural Information Processing Systems*, pages 5596–5605.

van den Hout, A. and van der Heijden, P. G. (2002). Randomized response, statistical disclosure control and misclassificatio: a review. *International Statistical Review*, **70**(2), 269–288.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, **9**(11).

Van Rooyen, B., Menon, A., and Williamson, R. C. (2015). Learning with symmetric label noise: The importance of being unhinged. In *Advances in Neural Information Processing Systems*, pages 10–18.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, **30**.

Wang, Y., Liu, W., Ma, X., Bailey, J., Zha, H., Song, L., and Xia, S.-T. (2018). Iterative learning with open-set noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8688–8696.

Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., and Bailey, J. (2019). Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 322–330.

Wei, H., Feng, L., Chen, X., and An, B. (2020). Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735.

Wei, H., Tao, L., Xie, R., and An, B. (2021a). Open-set label noise can improve robustness against inherent label noise. *NIPS*.

Wei, J., Liu, H., Liu, T., Niu, G., Sugiyama, M., and Liu, Y. (2021b). To smooth or not? when label smoothing meets noisy labels. *arXiv e-prints*, pages arXiv–2106.

Wu, X., Zhang, S., Zhou, Q., Yang, Z., Zhao, C., and Latecki, L. J. (2021). Entropy minimization versus diversity maximization for domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., and Sugiyama, M. (2019). Are anchor points really indispensable in label-noise learning? In *Advances in Neural Information Processing Systems*, pages 6838–6849.

Xia, X., Liu, T., Han, B., Wang, N., Gong, M., Liu, H., Niu, G., Tao, D., and Sugiyama, M. (2020). Part-dependent label noise: Towards instance-dependent label noise. *Advances in Neural Information Processing Systems*, **33**, 7597–7610.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. (2015). Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699.

Xu, Y., Cao, P., Kong, Y., and Wang, Y. (2019). L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise. *Advances in Neural Information Processing Systems*, **32**, 6225–6236.

Yi, K. and Wu, J. (2019). Probabilistic end-to-end noise correction for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7017–7025.

Yu, X., Liu, T., Gong, M., and Tao, D. (2018). Learning with biased complementary labels. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 68–83.

Yu, X., Han, B., Yao, J., Niu, G., Tsang, I., and Sugiyama, M. (2019). How does disagreement help generalization against label corruption? In *36th International Conference on Machine Learning, ICML 2019*.

Zagoruyko, S. and Komodakis, N. (2016). Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2018a). Understanding deep learning requires rethinking generalization.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018b). mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Zhang, Y., Zheng, S., Wu, P., Goswami, M., and Chen, C. (2020). Learning with feature-dependent label noise: A progressive approach. In *International Conference on Learning Representations*.

Zhang, Z. and Sabuncu, M. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788.

Zheng, S., Wu, P., Goswami, A., Goswami, M., Metaxas, D., and Chen, C. (2020). Error-bounded correction of noisy labels. In *ICML*, pages 11447–11457. PMLR.

Zhou, T., Wang, S., and Bilmes, J. (2020). Robust curriculum learning: From clean label detection to noisy label self-correction. In *International Conference on Learning Representations*.

Zhu, X. and Wu, X. (2004). Class noise vs. attribute noise: A quantitative study. *Artificial intelligence review*, **22**(3), 177–210.

Zhu, Z., Song, Y., and Liu, Y. (2021). Clusterability as an alternative to anchor points when learning with noisy labels. In *International Conference on Machine Learning*, pages 12912–12923. PMLR.