Meta Label Correction with Generalization Regularizer

Tao Tong , Yujie Mo , Yuchen Xie , Songyue Cai , Xiaoshuang Shi , Xiaofeng Zhu *
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
{tongqtao, moyujie2017, xyemrsnon, xsshi2013, seanzhuxf}@gmail.com, sonnycai@std.uestc.edu.cn

Abstract

Deep neural networks easily lead to the over-fitting issue due to the influence of noisy labels. However, previous label correction methods for dealing with noisy labels often need expensive computation cost to achieve effectiveness and ignore the generalization ability of the model. To address these issues, in this paper, we propose a new meta-based self-correction method to achieve accurate filtering of noisy labels and to enhance the generalization ability of the label correction model. Specifically, we first investigate a new gradient score method to filter noisy labels with less computation cost, and then theoretically design a new generalization regularizer into the meta-learner and the base learner, for correcting noisy labels as well as achieving the generalization ability. Experimental results on real datasets verify the effectiveness of our proposed method in terms of different classification tasks.

1 Introduction

The remarkable success of deep neural networks (DNNs) in various fields heavily depends on high-quality labels [Algan and Ulusoy, 2021; Wei *et al.*, 2023a; Sukhbaatar and Fergus, 2014; Arpit *et al.*, 2017]. However, many real-world datasets contain noisy labels for different reasons [Hospedales *et al.*, 2021]. For example, in medical datasets, even experienced professionals may occasionally produce noisy labels [Karimi *et al.*, 2020; Ju *et al.*, 2022]. Consequently, mitigating the impact of noisy labels is becoming a critical problem for researchers. Due to the fact that noisy labels can degrade model performance, reducing noise has become one of solutions for addressing the problem of learning with noisy labels. Among these, label correction is very popular since it substitutes original noisy labels with high-confidence pseudo labels to achieve significant model performance.

Label correction methods can be broadly categorized into two types based on the requirements for auxiliary information, *i.e.,* non-self-correction methods and self-correction methods. Existing non-self-correction methods often utilize auxiliary information to replace noisy labels with high-confidence pseudo labels. For example, [Ahn *et al.*, 2023] utilize a pre-trained model to learn effective feature representations for the generation of high-quality pseudo-labels. [Wu *et al.*, 2021] utilize soft labels of meta learning to correct noisy labels with clean samples. Self-correction methods integrate the self-iteration process of the model with deep neural networks to correct noisy labels, without using auxiliary information. For example, [Tanaka *et al.*, 2018] design a specialized network architecture to first estimate and then correct the distribution of noisy labels, while [Li *et al.*, 2022] design a noise correction loss function to correct noisy labels and [Li *et al.*, 2023] utilize noisy tolerant methods under the framework of meta learning to correct noisy labels. Due to the flexibility of noise correction rules, self-correction methods are more popular than non-self-correction methods in real-world applications.

Although previous label correction methods have achieved significant correction performance, there are still some limitations to be tackled. First, both non-self-correction and self-correction methods focus solely on minimizing losses during the optimization process, without considering overall generalization ability of the model. Consequently, this may lead to poor generalization performance. For example, the methods (such as Co-teaching [Yu *et al.*, 2019] and MLNT [Li *et al.*, 2023]) perform well on the training set, but their correction performance significantly decreases wit the increase of the noise ratio, resulting in low generalization ability. Second, current self-correction methods frequently employ loss-based methods (*i.e.,* Beta Mixture Model (BMM) [Arazo *et al.*, 2019] and Gaussian Mixture Model (GMM) [Permuter *et al.*, 2006]) to filter noisy labels, and thus requiring expensive computation resources and receiving limited filtering effectiveness for noisy labels.

To address above issues, we propose a new self-correction method, namely Meta Label Correction with Generalization Regularizer (MLCGR), as shown in Figure 1, to achieve accurate filtering of noisy labels and to enhance generalization ability. To do this, the proposed framework includes the filter and the corrector. The filter uses the gradient score (including the norm and the cosine angle between two adjacent gradients of every sample) to partition the whole dataset into three parts, resulting in accurate data partition and thus exploring the second issue of previous methods. The corrector designs a generalization regularizer into previous meta-learner and

*Corresponding authors (seanzhuxf@gmail.com).

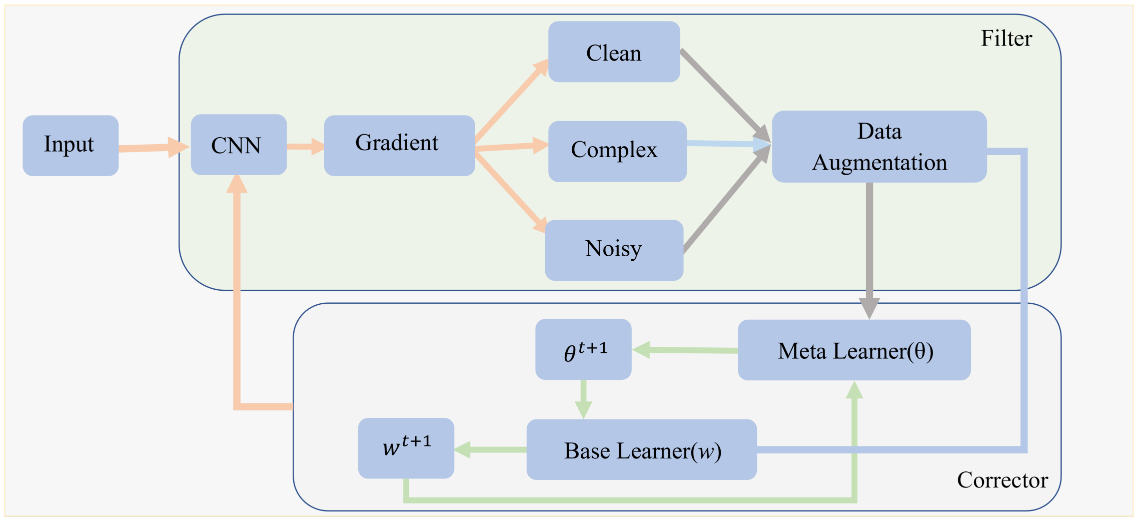


Figure 1: The framework of the proposed MLCGR, including two modules, *i.e.,* the filter and the corrector. Given the input $\mathbf{D}$, the filter first partitions $\mathbf{D}$ into three datasets, *i.e.,* the clean data $\mathbf{D}_{cl}$, the complex data $\mathbf{D}_{cp}$, and the noisy data $\mathbf{D}_{ny}$, based on the gradients of the model parameters, and then conducts data augmentation on these three datasets, which are augmented by different data augmentation methods. Specifically, we apply weak augmentation on the complex data and strong augmentation on both the clean data and the noisy data. The corrector trains the meta learner by the augmented clean data and the noisy data as well as trains the base learner by the augmented complex data. Moreover, the trained meta learner supervises the update of the base learner, followed by that the updated based learner supervises the update of the meta learner.

base-learner. Moreover, we theoretically demonstrate that our method really achieves the generalization ability, thus exploring the first issue of previous methods.

Compared to previous label correction methods, the main contributions of our method can be summarized as follows:

- The proposed method investigates a new method that allows the meta-based model to automatically correct noisy labels without manual specifying the meta-set.
- We theoretically explain the reasons that affect the generalization ability of the model, as well as design a new regularizer to achieve the generalization ability of the model.
- Extensive experiments on two synthetic datasets with different noise ratios and one real-world dataset validate the effectiveness of the proposed method compared to all comparison methods on classification tasks.

2 Method

Denoting $\mathbf{D} = \{\mathbf{X}, \mathbf{Y}\} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_1^n$ as the original dataset with n samples where $\mathbf{x}_i$ is the i-th training sample and $\mathbf{y}_i \in \{0,1\}^C$ is the corresponding label with C classes, denoting $\tilde{\mathbf{y}}$ and $\hat{\mathbf{y}}$, respectively, as the predicted label and the corrected label, the goal of label correction is to train a classification network $\mathbf{w}$ with the guidance of a label correction network $\boldsymbol{\theta}$.

2.1 Motivation

Meta-based methods for label correction attempt to replace noisy labels with high-confidence pseudo labels. For example, [Shu *et al.*, 2019] first use Meta Weight Net to relieve negative effect caused by noisy labels, and then assign different values to different losses. [Wu *et al.*, 2021] propose to replace noisy labels with high-confidence pseudo labels directly. However, these methods need to manually specify the clean data as the meta-set for the extraction of meta-knowledge. Unfortunately, the meta-set is difficult to obtain in real scenarios. Moreover, the scale of meta-set is quite essential for model performance. In addition, the optimization of existing works only consider the minimization of the loss by ignoring the impact of the generalization ability of the model. To address these issues, we propose the MLCGR method to achieve better correction and generalization performance, as shown in Figure 1.

2.2 Noisy Label Filtering

The goal of filtering methods is to distinguish noisy labels from others. Previous DNN methods tend to fit samples with clean labels fast by setting them with a small loss, compared to the samples with noisy labels [Wei *et al.*, 2023b]. For example, [Arazo *et al.*, 2019] use BMM to distinguish clean and noisy labels by estimating predicted confidence of every sample. [Li *et al.*, 2020] try to fit a two-component GMM to filter clean and noisy labels considering the flexibility in distribution prediction. Although these loss-based methods can effectively filter clean and noisy labels, they still have disadvantages. For example, in the early stage of the model training, clean labels can result in significant loss, but the model may lead to wrong filtering results. Besides, classic methods like GMM require extra memory to make the usage conditions stringent. To address these issues for achieving better filtering performance, we investigate the gradient-based measurement to filter noisy labels.

Given a dataset D with noisy labels, we first divide original dataset into multiple batches. Specifically, we use convolutional neural network (CNN) to train all samples and record their gradient scores ($\mathbf{G}$) during the training process, *i.e.,*

$$\mathbf{G} = \{(\mathbf{g}_1^1, \mathbf{g}_2^1, ..., \mathbf{g}_n^1), (\mathbf{g}_1^2, \mathbf{g}_2^2, ..., \mathbf{g}_n^2), ..., (\mathbf{g}_1^m, \mathbf{g}_2^m, ..., \mathbf{g}_n^m)\}, \quad (1)$$

where m is the number of epochs. To distinguish noisy labels, we consider the change in gradient scores as well as the variations in gradient angles, which is formulated by:

$$\begin{aligned} Score_i = Norm(\frac{1}{m-1}\sum_{j=2}^{m} \|\mathbf{g}_i^j - \mathbf{g}_i^{j-1}\|_2^2) \\ -\frac{\lambda_s}{m-1}\sum_{j=2}^{m} \cos(\mathbf{g}_i^j, \mathbf{g}_i^{j-1}), \quad i \in \{1, 2, ..., n\}, \end{aligned} \quad (2)$$

where λ_s is a non-negative parameter to balance two terms and $Norm$ is the normalization operator.

The first term in Eq. (2) calculates the norm of gradient variations of the same sample in adjacent rounds. We average all gradient variations and normalize them to make it match the second term. After that, the second term calculates the angle between two gradients of the same sample in two sequential epochs to prevent the occurrence of gradient variations, where the value of the norm changes slightly and the direction changes significantly. When all scores are sorted by their values, the higher the score is, the more likely its corresponding label is noisy. Furthermore, we use a non-negative parameter T $(0 < T < 1)$ to divide all samples into three subsets. Specifically, scores smaller than T will be considered as clean samples (*i.e.,* $\mathbf{D}_{cl}$), scores bigger than $1 - T$ will be considered as noisy samples (*i.e.,* $\mathbf{D}_{ny}$), and scores in between T and $1 - T$ will be considered as complex samples (*i.e.,* $\mathbf{D}_{cp}$). As a result, score calculation by Eq. (2) makes our method early capture the variations of the gradient norm in two consecutive epochs and the changes in the cosine similarity of two gradients in two consecutive epochs. Early detection of gradient variations leads to early detection of noisy labels. Therefore, noisy labels can be accurately filtered and the model requires less memory (verified in Section 3.3). In contrast, other noisy filtering methods, *i.e.,* GMM, filter noisy labels by modeling the distribution of the sample loss, making it difficult to detect noisy labels during the early stage of the training process.

After partitioning the inputted data into three subsets, *i.e.,* $\mathbf{D}_{cl}$, $\mathbf{D}_{cp}$, and $\mathbf{D}_{ny}$, by scoring gradient variations, the size of every subset is small. Hence, it is crucial to add the sample size. In this case, data augmentation has played an important role in meta-label correction. However, previous works generally treat all samples equally. For example, [Hussain *et al.*, 2017] use flipping to capture features that appear in various orientations. [Puttaruksa and Taeprasartsit, 2018] apply color jittering to help the model learn more invariant features. However, in these methods, data augmentation makes the samples with noisy labels amplify the noise information as well. In order to address this issue, we investigate to apply different methods of data augmentation to different subsets. Specifically, we apply weak augmentation with flipping and random cropping on complex data, as well as strong augmentation (such as AutoAugment [Cubuk *et al.*, 2018] and RandAugment [Cubuk *et al.*, 2020]) on both the clean data and the noisy data. After this, we obtain the final augmented data with the following form:

$$\begin{cases} \bar{\mathbf{D}}_{cl} = Aug_{strong}(\mathbf{D}_{cl}) \\ \bar{\mathbf{D}}_{cp} = Aug_{weak}(\mathbf{D}_{cp}) \\ \bar{\mathbf{D}}_{ny} = Aug_{strong}(\mathbf{D}_{ny}) \end{cases} . \quad (3)$$

In Eq. (3), weak augmentation preserves the fundamental characteristics of the data with noisy labels, but does not introduce significant changes. Hence, it mitigates the disruptive influence of noisy labels on model training. In contrast, strong augmentation enhances data diversity, compelling the model to learn robust features and thus improving its performance. Therefore, applying various augmentation strategies on different data is superior to equally treating all data.

2.3 Meta Label Correction with Generalization Regularizer

After data augmentation, each sample is expected to provide additional useful information for the downstream tasks. These augmented subsets (*i.e.,* $\bar{\mathbf{D}}_{cl}$, $\bar{\mathbf{D}}_{cp}$ and $\bar{\mathbf{D}}_{ny}$) are then sent to the meta-learner and the base-learner for noisy label correction. However, previous works typically minimize the loss function to obtain high-confidence pseudo labels, which are then used to replace original noisy labels. For example, both MLC and MSLC utilize the meta-learner to train a network of label correction with the help of soft labels and label smoothing. However, they ignore the generalization ability of the model, and thus they can not guarantee robustness of the downstream tasks. To solve this issue, we propose to correct noisy labels by taking the generalization ability of the model into account.

To achieve this, we first follow Model-agnostic Meta Learning (MAML) [Finn *et al.*, 2017], including the inner (*i.e.,* the base-learner $\mathcal{L}^{tr}$ with the parameter $\mathbf{w}$) and the outer (*i.e.,* meta-learner $\mathcal{L}^{me}$ with the parameter $\boldsymbol{\theta}$), to use the following cross-entropy loss ℓ:

$$\ell = -\sum_{\mathbf{x}\in\mathbf{X}}\sum_{i=1}^{C} y_i \log(f(\mathbf{x}_i; \boldsymbol{\theta}_\ell)), \quad (4)$$

where $\mathbf{y}_i$ denotes the ground truth label and $f(\mathbf{x}_i; \boldsymbol{\theta}_\ell)$ denotes the predict result. After this, the loss function of the meta-learning can be formulated as:

$$\begin{cases} \ell^{tr}(\mathbf{w}) = \ell(\mathbf{y}^{tr}, f(\mathbf{x}^{tr}, \mathbf{w})) & (5) \\ \ell^{me}(\boldsymbol{\theta}) = \ell(\mathbf{y}^{me}, f(\mathbf{x}^{me}, \mathbf{w}(\boldsymbol{\theta}))), & (6) \end{cases}$$

where $\boldsymbol{\theta}$ and $\mathbf{w}$ are the parameters of the meta-learner and the base-learner.

Since Eq. (5) only takes the original label $\mathbf{y}$ into account, it ignores the positive effect of predicted label $\tilde{\mathbf{y}}$. To address this issue, we use Eq. (7) to take both labels into account by setting $\hat{\mathbf{y}}^0 = \mathbf{y}$, where $\hat{\mathbf{y}}$ is the correct label and λ_y is the balance parameter.

$$\hat{\mathbf{y}}^{(t)} = \lambda_y \hat{\mathbf{y}}^{(t-1)} + (1 - \lambda_y)\tilde{\mathbf{y}}^{(t)}. \quad (7)$$

After that, Eq. (5) can be rewritten as follows:

$$\ell^{tr}(\mathbf{w}) = \ell(\hat{\mathbf{y}}^{cp}, f(\mathbf{x}^{cp}, \mathbf{w})). \quad (8)$$

In order to make better use of the information after data augmentation, we first regard noisy labels as negative information for the meter-learner, and then rewrite Eq. (6) as:

$$\begin{aligned}\ell^{me} = &- \sum_{(\mathbf{x},\mathbf{y})\in\bar{\mathbf{D}}_{cl}} \sum_{i=1}^{C} \mathbf{y}_i^{cl} \log(f(\mathbf{x}_i^{cl}, \boldsymbol{\theta})) \\ &- \lambda_{ny} \sum_{(\mathbf{x},\mathbf{y})\in\bar{\mathbf{D}}_{ny}} \sum_{i=1}^{C} \mathbf{y}_i^{ny} \log(1 - f(\mathbf{x}_i^{ny}, \boldsymbol{\theta})),\end{aligned} \tag{9}$$

where λ_{ny} balances the clean set and the noisy set. Eq. (9) introduces noisy label information into the loss function, which drives the predicted labels far away from incorrect ones to reduce the impact of noisy labels on model performance, and result in learning discriminative features.

Although Eq. (9) can handle the problem of noisy labels, it only focuses on the loss of the model and ignores the model generalization. In this paper, we propose a new regularizer to take the generalization ability of the model into account, *i.e.,*

$$\begin{aligned}&\underset{\varphi\in\Psi}{min}\, H(\boldsymbol{\Theta}) = \sum_{p,q\in y, p<q} 4^{\frac{z_1(\boldsymbol{\Theta})}{z_2^{(p,q)}(\boldsymbol{\Theta})}} \\ &s.t. \begin{cases} z_1(\boldsymbol{\Theta}) = \underset{\mathbf{u}}{min}\ \underset{\mathbf{x}_i\in\mathbf{X}}{max} \|\varphi(\mathbf{x}_i; \boldsymbol{\Theta}) - \varphi(\mathbf{X}; \boldsymbol{\Theta})\sigma(\mathbf{u})\|_2^2 \\ z_2(\boldsymbol{\Theta}) = \underset{\mathbf{v},\mathbf{s}}{max} \|\varphi(\mathbf{X}_p)\sigma(\mathbf{v}) - \varphi(\mathbf{X}_q)\sigma(\mathbf{s})\|_2^2 \end{cases},\end{aligned} \tag{10}$$

where $H(\boldsymbol{\Theta})$ is a function to calculate VC dimension (*i.e.,* d_{VC}) that can bound the generalization ability of the mode and φ is the feature extraction network with parameter $\boldsymbol{\Theta}$. In addition, $z_1(\boldsymbol{\Theta})$ is a function to calculate R which is the radius of a sphere, $z_2(\boldsymbol{\Theta})$ is a function to calculate M which is the distance between nearest sample and the hyperplane. The variables (such $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{s}$) are learnable vectors to help project samples on the sphere. The details of Eq. (10) will be explained in Supplementary Materials Section B.

2.4 Objective Function

By letting $\mathcal{L}^{VC} = H(\boldsymbol{\Theta})$ in Eq. (10) and making the expression concise, we set $e = \varphi(\mathbf{x}; \Theta)$, so our proposed loss function is formulated as follows.

$$\begin{cases} \mathcal{L}^{tr}(\mathbf{w}) = \ell^{tr}(\hat{\mathbf{y}}^{tr}, f(\mathbf{e}^{tr}, \mathbf{w})) + \gamma\mathcal{L}^{VC} \\ \mathcal{L}^{me}(\mathbf{w}^*(\boldsymbol{\theta})) = \ell^{me}(f(\mathbf{e}^{me}; \mathbf{w}^*(\boldsymbol{\theta})), \mathbf{y}^{me}) + \gamma\mathcal{L}^{VC} \end{cases}. \tag{11}$$

By minimizing the loss of $\mathcal{L}^{tr}$ and $\mathcal{L}^{me}$, we obtain the minimized $\mathcal{L}^{VC}$. Furthermore, small d_{VC} can tighten the upper bound of the generalization error, and thus achieving the generalization ability of the model. In this paper, we design a new bi-level minimization method to optimize our proposed objection function, *i.e.,*

$$\begin{cases} \mathbf{w}^*(\boldsymbol{\theta}) = \arg\min_{\mathbf{w}} \mathcal{L}^{tr}(\mathbf{w}; \boldsymbol{\theta}) \\ \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{me}(\mathbf{w}^*(\boldsymbol{\theta})) \end{cases}. \tag{12}$$

Specifically, we employ Stochastic Gradient Descent (SGD) [Amari, 1993] to alternately optimize Eq. (12). The adaptive optimization process is separated into three steps. In the first step, we extract a small batch of meta samples from $\mathbf{X}^{tr}$ as $\{(\mathbf{x}_i^{tr}, \mathbf{y}_i^{tr})\}_{i=1}^{n_1}$, where n_1 is the number of training samples, the parameter $\mathbf{w}$ can be updated by the following rule:

Algorithm 1 The pseudo-code of the proposed MLCGR.

```
Input: Training data $\mathbf{D} = \{\mathbf{X}, \mathbf{Y}\}$, the batch size, the maximal epoch (Max_E), and the maximal correction iteration (Max_C).
Output: the parameter of the target classifier network $w$.
 1: Initialize the parameters of the meta network, i.e., $w^{(0)}$ and $\boldsymbol{\theta}^{(0)}$, the filtering network $\boldsymbol{\theta}_f$, and the parameters, i.e., $\lambda_s$, $T$, $\lambda_y$, $\lambda_{ny}$, and $\gamma$.
 2: while $I < Max\_E$ do
 3:     $\mathbf{G} = WarmUp(\mathbf{D}, \boldsymbol{\theta}_f)$.
 4:     $\mathbf{D}_{cl}, \mathbf{D}_{cp}, \mathbf{D}_{ny} = Score(\mathbf{X}, \mathbf{Y}, \mathbf{G})$.
 5:     while $t < Max\_C$ do
 6:         $\{(\mathbf{x}_{cl}^{me}, \mathbf{y}_{cl}^{me}), (\mathbf{x}_{ny}^{me}, \mathbf{y}_{ny}^{me})\} = Batch(\mathbf{D}_{cl}, \mathbf{D}_{ny})$.
 7:         $\{(\mathbf{x}_{cp}^{tr}, \mathbf{y}_{cp}^{tr})\} = Batch(\mathbf{D}_{cp})$.
 8:         $\bar{\mathbf{x}}^{me} = Aug_{strong}(\mathbf{x}^{me}), \bar{\mathbf{x}}_{tr} = Aug_{weak}(\mathbf{x}_{tr})$.
 9:         Update $\boldsymbol{\theta}^{(t+1)}$ by Eq. (14).
10:         Update $w^{(t+1)}$ by Eq. (15).
11:         Update ($\tilde{\mathbf{y}}^{(t+1)}$, $\hat{\mathbf{y}}^{(t+1)}$ with parameters $\mathbf{w}^{(t+1)}$ and $\boldsymbol{\theta}^{(t+1)}$.
12:     end while
13: end while
```

$$\begin{aligned}\hat{\mathbf{w}}^{(t)} &= \mathbf{w}^{(t)} - \eta_1 \nabla_{\mathbf{w}} \mathcal{L}^{tr}(\mathbf{w}; \boldsymbol{\theta}) \\ &= \mathbf{w}^{(t)} - \frac{1}{n_1}\eta_1 (\sum_{i=1}^{n_1} \nabla_{\mathbf{w}} I_i^{tr})|_{\mathbf{w}^{(t)}},\end{aligned} \tag{13}$$

where $I_i^{tr} = \ell^{tr}(f(\mathbf{e}_i^{tr}; \mathbf{w}^{(t)}), \hat{\mathbf{y}}_i^{tr}(\boldsymbol{\theta}^{(t)}))$.

After updating the base-learner parameter $\mathbf{w}^{(t)}$ to $\hat{\mathbf{w}}^{(t)}$ through Eq. (13), in the second step, we update the parameter $\boldsymbol{\theta}$ of the mete-learner by:

$$\begin{aligned}\boldsymbol{\theta}^{(t+1)} &= \boldsymbol{\theta}^{(t)} - \eta_2 \nabla_{\theta} \mathcal{L}^{me}(\hat{\mathbf{w}}(\boldsymbol{\theta})) \\ &= \boldsymbol{\theta}^{(t)} - \frac{1}{n_2}\eta_2 (\sum_{i=1}^{n_2} \nabla_{\boldsymbol{\theta}} J_i^{me})|_{\boldsymbol{\theta}^{(t)}},\end{aligned} \tag{14}$$

where $J_i^{me} = \mathcal{L}^{me}(f(\mathbf{e}_i^{me}; \hat{\mathbf{w}}^{(t)}), \mathbf{y}_i^{me}) andandn_2$ is the number of training samples.

The update of $\mathbf{w}$ is under the help of unmodified label correction network, so it may lead to inaccurate pseudo labels. To address this issue, in the third step, we re-update $\mathbf{w}$ after the update of $\boldsymbol{\theta}$ (*i.e.,* $\boldsymbol{\theta}^{(t+1)}$), so the specific update rule for $\mathbf{w}^{(t+1)}$ is:

$$\begin{aligned}\mathbf{w}^{(t+1)} &= \mathbf{w}^{(t)} - \eta_1 \nabla_{\mathbf{w}} \mathcal{L}^{tr}(\mathbf{w}; \boldsymbol{\theta}^{(t+1)}) \\ &= \mathbf{w}^{(t)} - \frac{1}{n_1}\eta_1 (\sum_{i=1}^{n_1} \nabla_{\mathbf{w}} K_i^{tr})|_{\mathbf{w}^{(t)}},\end{aligned} \tag{15}$$

where $K_i^{tr} = \ell^{tr}(f(\mathbf{e}_i^{tr}; \mathbf{w}^{(t)}), \hat{\mathbf{y}}_i^{tr}(\boldsymbol{\theta}^{(t+1)}))$.

By the re-update process, the noise ratio in either the meta-learner or the base-learner can gradually reduce. Therefore, noisy labels can be corrected more accurately (verified in Section 3.4). Different from traditional meta label correction methods (such as MSLC and MLC) that solely focus on the smoothness of predict pseudo-labels and the prediction of soft classification outcomes, our proposed method utilizes the information in noisy labels as well as takes the model generalization into account. Obviously, our method achieves effectiveness as well as generalization ability. We list the pseudo-code of the proposed MLCGR in Algorithm 1.

3 Experiments

In this section, we conduct experiments on two synthetic datasets and one real-world dataset with different noisy ratios, compared our proposed MLCGR with seven comparison methods, with respect to image classification task in terms of accuracy (ACC).

3.1 Experimental Setup[1]

Datasets: We conduct experiments on two synthetic datasets (*i.e.,* CIFAR10 and CIFAR100 [Krizhevsky *et al.*, 2009]) and one real-world dataset (*i.e.,* Clothing1M [Xiao *et al.*, 2015]). Their specific information is summarized in Supplementary Meterials C.1. In our experiments, we follow [Zheng *et al.*, 2021] to corrupt original labels (*i.e.,* CIFAR10 and CIFAR100) with symmetric and asymmetric noise.

Comparison Methods: Seven comparison methods include three meta-based methods (*i.e.,* , MLC [Zheng *et al.*, 2021], MSLC [Wu *et al.*, 2021], and MLNT [Li *et al.*, 2023]), three traditional label correction methods (*i.e.,* Bootstrapping [Reed *et al.*, 2014], M-Correction [Arazo *et al.*, 2019], and Co-teaching [Han *et al.*, 2018]), and the baseline method, *i.e.,* Cross-Entropy [De Boer *et al.*, 2005].

Implementation Datails: We conduct all experiments on a computer with an Intel CPU Core(TM) i9-12900K @3.2 GHz 16-core and a NVIDIA GeForce RTX3090 to implement all methods including the proposed MLCGR with PyTorch framework. We obtain the source code of all comparison methods from the authors, and set the parameters by following the original literature, so that all comparison methods achieve their best performance in our experiments. In the proposed MLCGR, we apply ReLU as the activation function and SGD as the optimizer with a momentum of 0.9. We also let the learning rate as 10^{-2} and then gradually decay as 10^{-5}. For noisy filtering, we set the parameters λ_s and T as 0.3 and 0.1, respectively. For meta label correction, we set the label smooth parameter λ_y as 0.2, the noisy label parameter λ_{ny} as 0.1 and the generalization regularizer parameter γ as 0.1. The proposed MLCGR employs ResNet32 as the backbone to extract sample representation for the symmetric noisy and the ResNet-28-10 [Zagoruyko, 2016] for asymmetric noisy. For the real-world dataset Clothing1M, we follow the setting of previous work [Tanaka *et al.*, 2018], and use ResNet-50 a pre-trained model on ImageNet as the backbone. Moreover, we set the number of the warm-up epoch for CIFAR10, CIFAR100 and Clothing1M as 10, 20, and 50, respectively.

[1]Related Work and more details of Experimental Setup are reported in Supplementary Materials.

3.2 Result Analysis

We list the classification results of all methods in Tables 1 and 2. Obviously, the proposed MLCGR outperforms all comparison methods on all datasets, followed by Cross-Entropy, Bootstrapping, M-Correction, Co-teaching, MSLC, MLC and MLNT. For example, our method improves an average 3.41% and 13.86%, respectively, compared to the best comparison method (*i.e.,* M-correction) and the worst comparison method (*i.e.,* Cross-Entropy). This indicates that it is necessary to simultaneously filter noisy labels with gradient variations and correct them with the help of generalization regularizer for label correction methods.

First, compared to traditional label correction methods (*i.e.,* Cross-Entropy, Bootstrapping, M-Correction and Co-teaching), the proposed MLCGR outperforms them by a large margin. For example, the proposed MLCGR achieves an average improvement of 3.14% over the best correction method (*i.e.,* M-Correction) on all datasets. This verifies the superiority of meta-based label correction methods, as they have good adaptability to various ratios of noisy labels.

Second, compared to meta-based label correction methods (*i.e.,* MSLC, MLC and MLNT), the proposed MLCGR achieves an average improvement of 5.97% over the best correction method (*i.e.,* MLNT) on all datasets. This indicates that the meta-based methods for label correction improve the model performance by achieving the generalization ability of the model.

3.3 Filtering Effectiveness

We investigate the memory consumption and filtering accuracy of GMM on all datasets to evaluate the filtering effectiveness of the proposed MLCGR, and list the results in Figure 2. Obviously, our proposed MLCGR requires less memory and achieves higher filtering accuracy, compared to GMM.

Based on Figure 2a, MLCGR requires an average of 3149.88 KB less memory than GMM. In addition, with increasing epochs, MLCGR is relatively stable, while GMM increases significantly in terms of memory consumption. MLCGR's additional memory consumption scales with sample size, while GMM's is tied to the dimensionality of the data's covariance matrix.

Based on Figure 2b, MLCGR achieves an average increase of 13.62% in noisy label filtering accuracy. Moreover, with the increasing epochs, MLCGR is very stable, while GMM is with fluctuations, in terms of filtering noisy labels. The reason is that the MLCGR method achieves good performance for label filtering performance within a few epochs. In this case, GMM depending on the loss function cannot achieve good performance for label filtering because the loss of clean samples is high during the early states of model training.

3.4 Correction Effectiveness

We report the correction accuracy of all methods on two datasets (*i.e.,* CIFAR10 and CIFAR100) under 20% noisy ratio in Figure 3[2] Obviously, the proposed MLCGR outperforms all comparison methods on these two datasets.

[2]Figure 3 does not report the results of Dataset Clothing1M since it has no ground truth.

Noise Type	Symmetric							
Dataset	CIFAR10				CIFAR100			
Method /ratio	20%	40%	60%	80%	20%	40%	60%	80%
Cross-Entropy [De Boer *et al.*, 2005]	86.80	82.54	73.67	62.91	62.03	51.18	44.42	19.90
Bootstrapping [Reed *et al.*, 2014]	91.46	88.75	84.03	63.80	69.79	63.73	57.20	17.63
M-Correction [Arazo *et al.*, 2019]	93.02	92.47	89.36	86.82	73.90	68.53	59.32	48.20
Co-teaching [Han *et al.*, 2018]	89.54	87.21	77.93	67.41	65.60	58.46	44.32	27.90
MSLC [Wu *et al.*, 2021]	93.46	91.42	87.39	69.87	72.51	68.98	60.81	24.32
MLC [Zheng *et al.*, 2021]	92.63	90.04	86.21	77.41	66.80	61.39	47.21	21.80
MLNT [Li *et al.*, 2023]	92.95	91.35	84.48	74.39	68.50	64.29	55.73	42.40
MLCGR (Proposed)	**95.32**	**94.87**	**93.57**	**92.34**	**75.36**	**71.09**	**63.47**	**48.27**

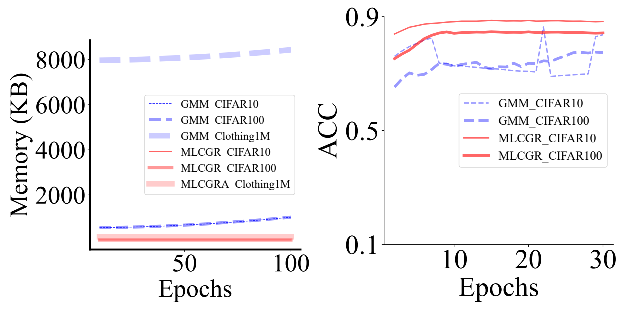
Table 1: Classification accuracy of all methods on two datasets (*i.e.,* CIFAR10 and CIFAR100) at different symmetrically noisy ratios, where the best results are highlighted in bold.

Noise Type	Asymmetric								Real-world
Dataset	CIFAR10				CIFAR100				Clothing1M
Method /ratio	20%	40%	60%	80%	20%	40%	60%	80%	–
Cross Entropy [De Boer *et al.*, 2005]	92.85	90.22	67.21	57.46	69.05	65.14	40.59	15.83	68.94
Bootstrapping [Reed *et al.*, 2014]	93.08	91.18	87.24	81.07	70.93	67.82	44.53	15.85	69.12
M-Correction [Arazo *et al.*, 2019]	89.36	87.45	87.59	67.78	72.41	69.54	60.51	16.06	71.00
Co-teaching [Han *et al.*, 2018]	89.32	86.24	70.43	58.23	71.73	66.35	45.84	16.99	73.33
MSLC [Wu *et al.*, 2021]	94.39	92.81	84.14	64.23	72.66	70.51	58.35	17.42	73.47
MLC [Zheng *et al.*, 2021]	93.24	91.63	83.22	61.25	68.35	64.24	57.61	17.29	73.47
MLNT [Li *et al.*, 2023]	92.14	89.24	82.93	60.52	70.63	68.57	58.03	17.16	75.78
MLCGR(Proposed)	**96.04**	**94.69**	**90.36**	**84.54**	**73.14**	**71.06**	**60.54**	**17.82**	**75.83**

Table 2: Classification accuracy of all methods on two synthetic datasets (*i.e.,* CIFAR10 and CIFAR100) at different asymmetrically noisy ratios and one real-world dataset Clothing1M, where the best results are highlighted in bold.

NLF	GR	CIFAR10				CIFAR100				Clothing1M
		20%	40%	60%	80%	20%	40%	60%	80%	
–	–	84.25	80.31	72.59	64.59	67.24	59.81	45.36	18.48	68.25
✓	–	90.72	86.21	81.64	75.29	71.49	64.08	52.91	27.42	70.24
–	✓	93.75	92.43	91.54	90.85	73.56	69.47	59.26	45.31	73.28
✓	✓	**95.32**	**94.87**	**93.57**	**92.34**	**75.36**	**71.09**	**63.47**	**48.27**	**75.83**

Table 3: Classification accuracy of each module (*i.e.,* Noisy Label Filtering (NLF) and Generalization Regularizer (GR)) of our ML-CGR on two datasets (*i.e.,* CIFAR10 and CIFAR100) under different symmetrically noisy ratios.



(a) Extra Memory (b) Filtering Accuracy

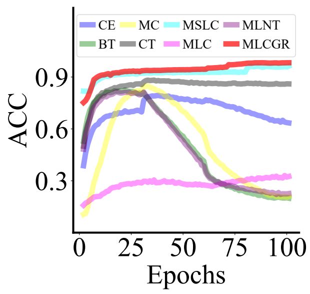
Figure 2: Memory consumption and filtering accuracy of MLGCR and GMM under 20% symmetric noise on all datasets.

For example, our method averagely improves by 5.8% and 53.86%, respectively, compared to the best comparison method (*i.e.,* Co-teaching) and the worst comparison method (*i.e.,* MLC). This verifies that our method achieves the best correction accuracy due to its generalization ability. In particular, our method does not achieve good classification performance in the first epochs, such as the first 15 epochs for Dataset CIFAR10 and the first 75 epochs for Dataset CIFAR100. After that, our method achieves good performance. The reason is that 1) in the first epochs our method has a limited number of clean samples to output bad correction accuracy and 2) our method with the increasing clean samples improves the generalization ability of the model with the increasing epochs.

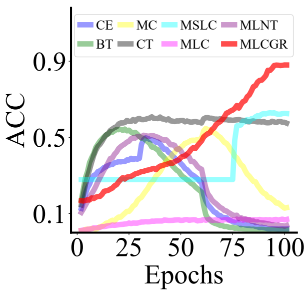
3.5 Ablation Study

The proposed method includes two key components, *i.e.,* noisy label filtering (NLF) in the filter module and the generalization regularizer (GR) in the corrector module. We demonstrate the effectiveness of each component by reporting the classification results of three methods (*i.e.,* NLF without GR in our method, GR without NLF in our method and our proposed MLCGR) in Table 3.

First, the proposed method considering two modules improves on average by 17.71%, compared to the method without considering any module. Moreover, the method without considering any module is worse than the methods consid-

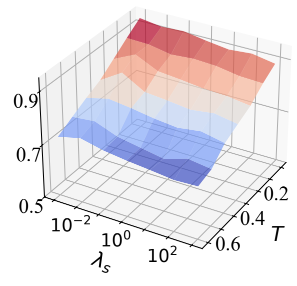


(a) CIFAR10

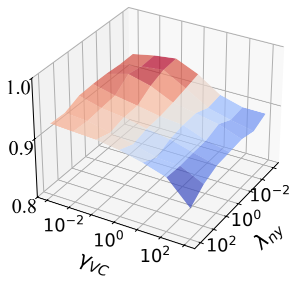


(b) CIFAR100

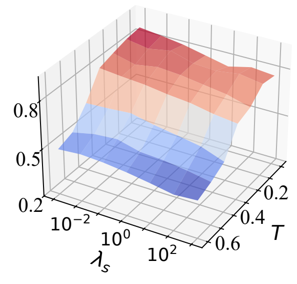
Figure 3: Correction accuracy of all methods on Datasets CIFAR10 and CIFAR100 under 20% symmetrical noise (CE is Cross-Entropy, BT is Bootstrapping, MC is M-Correction, CT is Co-Teaching).



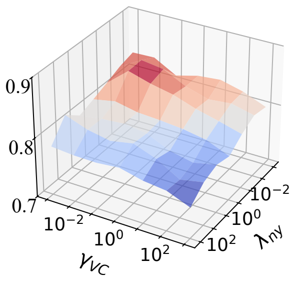
(a) λ_s and T



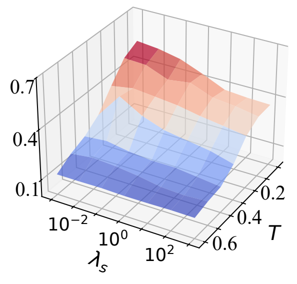
(b) λ_s and T



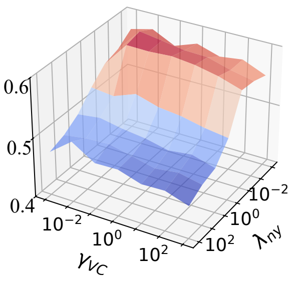
(c) λ_s and T



(d) λ_{ny} and γ



(e) λ_{ny} and γ



(f) λ_{ny} and γ

Figure 4: Classification results of MLCGR under different parameter settings (*i.e.,* λ_s, T, λ_{ny} and γ) on CIFAR10 (top), CIFAR100 (middle) and Clothing1M (bottom).

ering only one module, *i.e.,* NLF and GR. This verifies that NLF or GR is useful for label correction. In particularly, our method outperforms NLF or GR. This demonstrates that we should consider both NLF and GR for label correction. Second, in two modules, NLF outperforms GR. For example, NLF improves on average about 7.71%, compared to GR, on all datasets. The reason is that accurate filtering and corresponding adaptive augmentation can result in the meta-set with the large sample size. Based on Lemma B.1 in the Supplementary Materials, the increasing samples can reduce the upper bound of the generalization error. Hence, NLF is able to improve the generalization ability of the model.

3.6 Parameter Analysis

In Eq. (2), we employ parameters λ_s and T, respectively, to achieve a trade-off between gradient norm and direction, as well as to divide all samples into three subsets. In experiments, we investigate their variations by setting $\lambda_s \in [10^{-3}, 10^3], T \in [0.1, 0.7]$, $\lambda_{ny} = 0.1$ and $\gamma = 0.1$ and report the results in the upper row of Figure 4. Obviously, our MLCGR obtains good performance if λ_s is smaller than 0.1 and T is smaller than 0.3. However, if the value of λ_s is greater than 1 and T is greater than 0.5, our MLCGR achieves bad performance. This demonstrates that large values of T and λ_s, respectively, result in more noise labels and recusing the filtering accuracy, and thus affecting the label correction.

In Eq. (11), we employ parameters λ_{ny} and γ, respectively, to enrich the training information and control the generalization ability. In experiments, we investigate their variations by setting $\lambda_{ny} \in [10^{-3}, 10^3]$, $\gamma \in [10^{-3}, 10^3$, $\lambda_s = 0.1$ and $T = 0.1$. We report the results in the bottom row of Figure 4. Obviously, the proposed MLCGR delivers superior performance when λ_{ny} and γ are smaller than 10^{-1}. However, if the values of λ_{ny} and γ are greater than 10^0, the proposed MLCGR cannot achieve satisfactory performance. This indicates that the effectiveness of the Cross-Entropy loss may be effected with the large values of λ_{ny} and γ, thus failing to provide sufficient high confidence labels for the model.

4 Conclusion

In this paper, we proposed an effective meta-based label correction method to address the issues in previous methods. Specifically, we investigated a filter to first partition the entire dataset to three subgroups based on the gradient scores and then to adaptively conduct data augmentation on every subgroups. for achieving better filtering subsets. We also deigned a new generalization regularizer into both the meta-learner and the base-learner to achieve the generalization capability of the model. Experimental results on real data sets verify the effectiveness of the proposed method in terms of different classification tasks in different noise ratios.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFA1004100, the National Natural Science Foundation of China under Grant 62306066.

References

[Ahn *et al.*, 2023] Sumyeong Ahn, Sihyeon Kim, Jongwoo Ko, and Se-Young Yun. Fine tuning pre trained models for robustness under noisy labels. In *IJCAI*, pages 3643–3651, 2023.

[Algan and Ulusoy, 2021] Görkem Algan and Ilkay Ulusoy. Image classification with deep learning in the presence of noisy labels: A survey. *Knowledge Based System.*, 215:106771–106790, 2021.

[Amari, 1993] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993.

[Arazo *et al.*, 2019] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *ICML*, pages 312–321, 2019.

[Arpit *et al.*, 2017] Devansh Arpit, Stanis law Jastrz kebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *ICML*, pages 233–242, 2017.

[Cubuk *et al.*, 2018] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv:1805.09501*, pages 1–14, 2018.

[Cubuk *et al.*, 2020] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. *CVPR*, pages 702–703, 2020.

[De Boer *et al.*, 2005] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135, 2017.

[Han *et al.*, 2018] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, pages 1–11, 2018.

[Hospedales *et al.*, 2021] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5149–5169, 2021.

[Hussain *et al.*, 2017] Zeshan Hussain, Francisco Gimenez, Darvin Yi, and Daniel Rubin. Differential data augmentation techniques for medical imaging classification tasks. In *AMIA*, volume 2017, pages 979–985, 2017.

[Ju *et al.*, 2022] Lie Ju, Xin Wang, Lin Wang, Dwarikanath Mahapatra, Xin Zhao, Quan Zhou, Tongliang Liu, and Zongyuan Ge. Improving medical images classification with label noise using dual-uncertainty estimation. *IEEE Transactions on Medical Imaging*, 41(6):1533–1546, 2022.

[Karimi *et al.*, 2020] Davood Karimi, Haoran Dou, Simon K Warfield, and Ali Gholipour. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, 65:101759–101808, 2020.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *arXiv*, 2009.

[Li *et al.*, 2020] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICML*, pages 1–14, 2020.

[Li *et al.*, 2022] Lin Li, Long Chen, Yifeng Huang, Zhimeng Zhang, Songyang Zhang, and Jun Xiao. The devil is in the labels: Noisy label correction for robust scene graph generation. In *CVPR*, pages 18869–18878, 2022.

[Li *et al.*, 2023] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *CVPR*, pages 5051–5059, 2023.

[Permuter *et al.*, 2006] Haim Permuter, Joseph Francos, and Ian Jermyn. A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern Recognition*, 39(4):695–706, 2006.

[Puttaruksa and Taeprasartsit, 2018] Chanachai Puttaruksa and Pinyo Taeprasartsit. Color data augmentation through learning color-mapping parameters between cameras. In *JCSSE*, pages 1–6, 2018.

[Reed *et al.*, 2014] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, pages 1–11, 2014.

[Shu *et al.*, 2019] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, volume 32, 2019.

[Sukhbaatar and Fergus, 2014] Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153, 2014.

[Tanaka *et al.*, 2018] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, pages 5552–5560, 2018.

[Wei *et al.*, 2023a] Jiaheng Wei, Zhaowei Zhu, Tianyi Luo, Ehsan Amid, Abhishek Kumar, and Yang Liu. To aggregate or not? learning with separate noisy labels. In *SIGKDD*, pages 2523–2535, 2023.

[Wei *et al.*, 2023b] Qi Wei, Lei Feng, Haoliang Sun, Ren Wang, Chenhui Guo, and Yilong Yin. Fine-grained classification with noisy labels. In *CVPR*, pages 11651–11660, 2023.

[Wu *et al.*, 2021] Yichen Wu, Jun Shu, Qi Xie, Qian Zhao, and Deyu Meng. Learning to purify noisy labels via meta soft label corrector. *AAAI*, pages 10388–10396, 2021.

[Xiao *et al.*, 2015] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, pages 2691–2699, 2015.

[Yu *et al.*, 2019] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *ICML*, pages 7164–7173, 2019.

[Zagoruyko, 2016] Sergey Zagoruyko. Wide residual networks. *arXiv:1605.07146*, pages 1–15, 2016.

[Zheng *et al.*, 2021] Guoqing Zheng, Ahmed Hassan Awadallah, and Susan Dumais. Meta label correction for noisy label learning. In *AAAI*, pages 11053–11061, 2021.