Push the Student to Learn Right:
Progressive Gradient Correcting by Meta-learner on Corrupted Labels

Jun Shu 1  Qi Xie 1  Lixuan Yi 1  Qian Zhao 1  Sanping Zhou 2  Zongben Xu 1  Deyu Meng* 1

Abstract
While deep networks have strong fitting capability to complex input patterns, they can easily overfit to biased training data with corrupted labels. Sample reweighting strategy is commonly used to alleviate this robust learning issue, through imposing zero or possibly smaller weights to corrupted samples to suppress their negative influence to learning. Current reweighting algorithms, however, need elaborate tuning of additional hyper-parameters or careful designing of complex meta-learner for learning to assign weights on samples. To address these issues, we propose a new meta-learning method with few tuned hyper-parameters and simple structure of a meta-learner (one hidden layer MLP network). Guided by a small amount of unbiased meta-data, the parameters of the proposed meta-learner can be gradually evolved for finely tugging the classifier gradient approaching to the right direction. This learning manner complies with a real teaching progress: A good teacher should more respect the student’s own learning manner and help progressively correct his learning bias based on his/her current learning status. Experimental results substantiate the robustness of the new algorithm on corrupted label cases, as well as its stability and efficiency in learning.

The dream begins with a teacher who believes in you, who tugs and pushes and leads you to the next plateau, sometimes poking you with a sharp stick called truth.
— Dan Rather

1. Introduction
Deep neural networks (DNNs) have recently obtained impressive good performance on various applications due to their powerful capacity for modeling complex input patterns. Despite their success, deep networks can easily overfit to biased training data containing corrupted labels (Zhang et al., 2017), leading to their poor performance in generalization in such cases. This issue has been theoretically illustrated in multiple current literatures (Neyshabur et al., 2017; Arpit et al., 2017; Kawaguchi et al., 2017; Novak et al., 2018).

In practice, however, such corrupted label issue is always encountered due to the high cost of labor and time to obtain high quality of data annotations (Sukhbaatar & Fergus, 2015; Azadi et al., 2016; Goldberger & Ben-Reuven, 2017; Li et al., 2017; Vahdat, 2017; Hendrycks et al., 2018; Han et al., 2018; Zhang & Sabuncu, 2018). A typical example is a dataset collected from a crowdsourcing system (Bi et al., 2014) or search engines (Liang et al., 2016; Zhuang et al., 2017), which would possibly yield a large amount of noisy labels. Effective learning with such corrupted labels, which can be regarded to be biased from ground-truth ones, is thus an important while challenging issue in machine learning (Jiang et al., 2018; Ren et al., 2018).

The sample weighting approach is commonly utilized against this issue. The main methodology is to design individual weighting schemes on samples based on specific tasks and models. The early attempts include the dataset resampling (Chawla et al., 2002) and instance reweight (Zadrozny, 2004) methods, imposing sample weights by making use of proper prior knowledge and minimizing the weighted loss on training samples. After that, multiple reweighting methods have been presented via dynamically updating sample weight through the learning process to ameliorate such preset weight manner. The clue is mainly achieved from the loss values of samples in training. The approach can be divided into two categories. One more emphasizes the samples with larger loss values since they are more like to be uncertain hard samples located on the classification boundary containing more information for distinguishing classes. Typical methods include AdaBoost (Freund & Schapire, 1997; Sun et al., 2007), hard negative mining (Malisiewicz et al., 2011) and focal loss (Lin et al.,...
The other takes the samples with smaller loss values as more important since they intend to be high-confident with clean labels, while suppress the effects of those with extremely large loss values. Typical methods include self-paced learning (Kumar et al., 2010), iterative reweighting (Fernando & Michael, 2003; Zhang & Sabuncu, 2018) and multiple variants (Jiang et al., 2014a; b; Wang et al., 2017). Such re-weighting strategies enhance the flexibility of sample weight evaluation and extend the feasible domains beyond the initial sample weight schemes. These method, however, still need certain assumptions for constructing models, inevitably involving hyper-parameters required to be manually preset or tuned by cross-validation. This tends to raise their application difficulty and reduce their performance stability in handling real problems.

Very recently, the meta-learning regime becomes a new trend for this issue. The basic idea is first to pre-collect a small amount of unbiased meta-data with clean labels to simulate the underlying meta-knowledge of the true sample-label distribution, and then design a meta-learner on the basis of such meta-data beyond the learner constructed on the original much larger biased training dataset with noisy labels. The sample weights can then be determined by meta-learner when inputting sample loss on learner. In this meta-learning paradigm, the tuning of hyper-parameters becomes automatic and is embedded into the learning process. However, to guarantee a strong capability of hyper-parameter learning, these methods generally need to construct a complex structure of meta-learner. The representative examples include the FWL (Dehghani et al., 2018), learning to teach (Fan et al., 2018; Wu et al., 2018) and MentorNet (Jiang et al., 2018) methods, whose meta-learners are designed as a Bayesian function approximator, a deep neural network with attention mechanism, and a bidirectional LSTM network, respectively. This makes the algorithms of these meta-learning methods hard to be fully re-produced and understood by general users.

To alleviate this issue, this paper proposes a new meta-learning method. The main idea is to progressively rectify the sample weights by the meta-learner so as to gradually correct the gradient of the weighted loss computed on the biased training set approaching to the descent direction of the loss calculated on the unbiased meta-data. In summary, the method has the following three-fold specific characteristics.

Firstly, the meta-learner used in our method, called V-Net, is an MLP network with only one hidden layer, whose form is extremely simpler as compared with those used in other meta-learning methods. All updating equations for the parameters of the meta-learner and the classifier are in closed form and explicitly expressed (as illustrated in Fig. 1). The code of the algorithm is thus easy to be re-produced.

Secondly, the working mechanism of the proposed meta-learning method finely complies with the education process of a student by a good teacher, which can be properly described as the quote cited in the beginning of the paper, said by the famous journalist and anchor, Dan Rather. Specifically, the teacher helps progressively correct the learning manner of the student (i.e., the gradient of the biased training loss), under the guidance of meta-knowledge (i.e., unbiased meta-data) only possessed by teachers, to the right direction oriented to the truth (i.e., the descent direction of the meta-loss). In this progressive learning process, the teacher and the student are collaborated to be ameliorated from each other (i.e., both parameters of the meta-learner and the classifier are gradually optimized in the learning process). We expect such intuitive explanations could make the idea of our method easily understood by common readers.

Thirdly, the insights of why the proposed algorithm can work can be well interpreted. Particularly, the closed-form updating equation for the parameter of the meta-learner can be properly explained to improve the sample weights of those samples better complying with the meta-data knowledge, while suppress weights of those violating such meta-knowledge. This tallies with our common sense on the problem: we should emphasize samples unbiased to the underlying groundtruth label distribution, while reduce the influence of those highly biased ones.

Experimental results substantiate the robustness of the proposed method on corrupted labels. Especially, we depict that the sample weights can be ameliorated in a stable manner, which makes our algorithm better memory historical learning knowledge underlying both the sample weights as well as the parameters of the meta-learner and the classifier.

The paper is organized as follows. Section 2 introduces related works. Section 3 presents the proposed meta-learning method as well as the detailed algorithm and analysis of its convergence property. Section 4 demonstrates experimental results and the conclusion is finally made.

2. Related Work

Sample Weighting Methods. The idea of weighting training samples can be dated back to dataset resampling (Chawla et al., 2002) or instance re-weight (Zadrozny, 2004), which pre-evaluate the sample weights as a preprocessing step by using certain prior knowledge on the task or data. To make the method more flexible and the sample weights better fit data, more recent researches focused on the dynamic sample re-weighting regimes to ameliorate weights during training process. Typical re-weighting methods include the boosting algorithm and multiple of its variations, like the known AdaBoost method (Freund & Schapire, 1997), which dynamically emphasizes relatively harder examples and imposes larger weights to ones with
larger loss values. Hard example mining (Malisiewicz et al., 2011) and focal loss (Lin et al., 2018) also enhance the functions of harder examples and put larger weights on them. On the contrary, another series of methods, like self-paced learning (Kumar et al., 2010), put more focus on easy samples with smaller losses. Afterwards, multiple extensions on self-paced learning are presented by incorporating with more knowledge of practical tasks (Jiang et al., 2014a;b). Iterative reweighting strategy (Fernando & Mkchael, 2003; Zhang & Sabuncu, 2018) also inclines to emphasize the functions of easy samples while suppress those of hard ones, possibly with corrupted label noises. Some other methods along this line include the prediction variance method (Chang et al., 2017), emphasizing more on uncertain samples to improve mini-batch SGD for classification, and the method given in (Wang et al., 2017), inferring the example weights as latent variables under an elaborately designed Bayesian framework. Although these methods more or less alleviate the corrupted label noise issue, they still need certain assumptions for designing their methods, naturally involving hyper-parameters required to be manually preset or tuned by the cross-validation. This, however, raises their difficulty to be readily used in real applications.

**Meta Learning Methods.** Inspired by recent meta-learning developments for few-shot learning (Lake et al., 2015; Shu et al., 2018; Ravi & Larochelle, 2017; Finn et al., 2017; Snell et al., 2017), where only a handful of examples are available for predicting classes, recently some meta-learning regimes are also proposed for the issue to make the hyper-parameter learning more automatic and reliable. Typical methods along this line include the FWL (Delghani et al., 2018), learning to teach (Fan et al., 2018; Wu et al., 2018) and MentorNet (Jiang et al., 2018) methods, whose meta-learners are designed as a Bayesian function approximator, a deep neural network with attention mechanism, a bidirectional LSTM network, respectively. Another method called L2RW (learning to reweight) (Ren et al., 2018), does not explicitly assume meta-learner while directly updates weights from zeros in each meta-learning iteration. For the former three methods, the meta-learners are with relatively more complex structures and their insights are hard to be fully interpreted. Comparatively, our method is with a much simplest form of meta-learner compared with them, facilitating its easier code re-production and understanding by users. The latter L2RW method, however, does not set meta-learner to guide the weight learning, which tends to make it possess the unstable learned weights across the entire learning process and slower convergence speed. Comparatively, our method can learn weights in a more constant manner, which makes the method convergent in less iterations, as clearly shown in Fig. 3 and 4.

**Other Methods for Corrupted Labels.** Learning with corrupted labels has attracted increasing attention recently both on theory and practice (Natarajan et al., 2013; Van Rooyen & Williamson, 2018). For literature comprehensiveness, we also list some other methods proposed for this task. Multiple methods are designed by correcting noisy labels to their true labels via a supplementary clean label inference step (Azadi et al., 2016; Vahdat, 2017; Veit et al., 2017; Li et al., 2017; Jiang et al., 2018; Ren et al., 2018; Hendrycks et al., 2018). A typical method is Li et al. (2017), to distill the knowledge from clean labels and knowledge graph, and can be exploited to learn a better model from noisy labels. More recently, GLC (Hendrycks et al., 2018) proposed a loss correction approach to mitigate the effects of label noise on deep neural network classifiers. Some other methods are also constructed by using proper methods to directly learn from corrupt labels. Typical ones include the Reed (Reed et al., 2015), Co-training (Han et al., 2018), D2L (Ma et al., 2018), S-Model (Goldberger & Ben-Reuven, 2017) methods. We will compare these methods in our experiments to make the superiority of the proposed method convincible.

### 3. The Proposed Meta-learning Method

In this section, we introduce the methodology of the proposed progressive gradient correcting method by meta-learner (Meta-PGC in brief), and its algorithm as well as its convergence analysis. To make all steps of the algorithm easily understandable, we introduce its implementations along with their corresponding interpretations from the perspective of the education process of a student by a teacher.

#### 3.1. The Meta-PGC Method

Consider a classification problem with the training set \( \{x_i, y_i\}_{i=1}^{N} \), where \( x_i \) denotes the \( i \)-th sample, \( y_i \in \{0, 1\}^c \) is the noisy label vector over \( c \) classes, and \( N \) is the number of the entire training data. \( f(x, w) \) denotes the classifier required to be achieved, where \( w \) denotes the model parameters. In current applications, \( f(x, w) \) is always set as a network. We thus also adopt the network structure for the classifier, and call it the student network for convenience in the following.

Generally, the optimal classifier parameter \( w^* \) can be extracted by minimizing the loss \( \frac{1}{N} \sum_{i=1}^{N} l(y_i, f(x_i, w)) \) calculated on the training set. For notation convenience, we denote that \( L_i^{\text{train}}(w) = l(y_i, f(x_i, w)) \).

In the presence of corrupted labels, sample re-weighting methods enhance the robustness of training by imposing weight \( g(x_i; \Theta) \) on the \( i \)-th sample loss, where \( \Theta \) represents the hyper-parameters contained in the weighting function. The optimal parameter \( w \) of the student network
is calculated by the following weighted loss minimization:

$$w^*(\Theta) = \arg\min_w \frac{1}{N} \sum_{i=1}^{N} g_c(L_i^{train}(w); \Theta) L_i^{train}(w).$$

(1)

**V-Net:** Instead of manual presetting, our method aims to automatically learn the hyper-parameter $\Theta$ in a meta-learning manner. To this aim, we formulate $g_c(L_i^{train}(w); \Theta)$ as a neural network, called V-Net, as a MLP network with only one hidden layer containing 100 nodes (i.e., the network is with a 1-100-1 structure), as shown in Fig. 1. Each hidden node is with ReLU activation function, and the output is with the Sigmoid activation function, to guarantee the output located in the interval of $[0, 1]$. As compared with meta-learners designed in current meta-learning methods (Dehghani et al., 2018; Fan et al., 2018; Wu et al., 2018; Jiang et al., 2018), such V-Net is with simpler form and less parameters.

**Meta learning.** The hyper-parameters contained in V-Net can be optimized by using the meta learning idea (Wu et al., 2018; Andrychowicz et al., 2016; Dehghani et al., 2017; Franceschi et al., 2018). Specifically, assume that we have a small amount unbiased meta-data set (i.e., with clean labels) $\{x_i^{(meta)}, y_i^{(meta)}\}_{i=1}^{M}$, representing the meta-knowledge of ground-truth sample-label distribution, where $M$ is the number of meta-samples and $M \ll N$. The optimal hyper-parameter can then be obtained by minimizing the meta-loss calculated on meta-data as follows:

$$\Theta^* = \arg\min_{\Theta} \frac{1}{M} \sum_{i=1}^{M} L_i^{meta}(w^*(\Theta)).$$

(2)

where $L_i^{meta}(w) = l(y_i^{(meta)}, f(x_i^{(meta)}, w))$.

Analogy with the education process, we can understand Eq. (1) as the learning manner of a student, the weight function (i.e., V-Net) as the the correction way for such learning manner imposed by a teacher (emphasizing more on valuable samples), Eq. (2) as the self-rectifying process of the teacher (i.e., the meta-learner) based on the teaching effect feeded back from the student.

**Formulating learning manner of student network:** To get the updating equation for the parameter $w$ of the student network, we need to calculate the gradient of the training loss objective in Eq. (1) so as to ameliorate the student network by gradient descent. As general network training tricks, we also employ SGD for this training task. Specifically, in each iteration of training, a mini-batch of training samples $\{(x_i, y_i), 1 \leq i \leq n\}$ is sampled, where $n$ is the mini-batch size, $n \ll N$. Then the updating equation of the student network parameter can be formulated by moving the current $w^{(t)}$ along the descent direction of the objective loss in Eq. (1) on a mini-batch training data. i.e.,

$$\hat{w}^{(t)}(\Theta) = w^{(t)} - \frac{1}{n} \times \sum_{i=1}^{n} g_c(L_i^{train}(w^{(t)}); \Theta) \nabla_w L_i^{train}(w) \bigg|_{w^{(t)}}.$$  

(3)

where $\alpha$ is the step size.

In perspective of education, this step can be explained as that the student attempts to explore the optimal learning manner $\hat{w}^{(t)}(\Theta)$ for achieving expected knowledge purely on training data based on his/her previous learning experience $w^{(t)}$. Note that the learning manner is parameterized by the hyper-parameter $\Theta$ of the teacher, which will be feeded back to teacher and further ameliorated by the teacher.

**Updating parameters of V-Net:** After receiving the feedback of the model parameter updating formulation $\hat{w}^{(t)}(\Theta)$ from the last step, the parameter $\Theta$ of the V-Net can then be readily updated guided by Eq. (2), i.e., moving the current parameter $\Theta^{(t)}$ along the objective gradient of Eq. (2) calculated on the meta-data. Similar to the last step, the SGD scheme is adopted. The updating equation for $\Theta$ is then:

$$\Theta^{(t+1)} = \Theta^{(t)} - \beta \frac{1}{m} \sum_{i=1}^{m} \nabla_{\Theta} L_i^{meta}(\hat{w}^{(t)}(\Theta)) \bigg|_{\Theta^{(t)}}.$$

(4)

where $\{x_i^{(meta)}, y_i^{(meta)}\}_{i=1}^{m}$ denote a mini-batch meta-data set calculated in the current step.

This step can be explained as that the teacher learns the right way for rectifying the learning manner of the student based on the meta-data knowledge. Just as Dan said: the teacher tugs/pushes/leads the student to the next plateau, poking the student with a stick of truth.

**Updating parameters of student network:** Then, the weights can be readily passed to the updating equation (3) to correct the gradient for ameliorating the parameter $w$ of the student network, i.e.,

$$w^{(t+1)} = w^{(t)} - \frac{1}{n} \times \sum_{i=1}^{n} g_c(L_i^{train}(w^{(t)}); \Theta^{(t+1)}) \nabla_w L_i^{train}(w) \bigg|_{w^{(t)}}.$$  

(5)

This can be easily explained as that the teacher helps to tug the student’s learning to the right direction and make him/her achieve better learning effect in the process.

### 3.2. Analysis on the Weighting Scheme of V-Net

The computation of Eq. (3) and Eq. (5) can be easily implemented by automatic differentiation, and the computation
of Eq. (4) can be tackled by following derivation:

\[
\frac{1}{m} \sum_{i=1}^{m} \nabla_{\Theta} L_{i}^{\text{meta}}(\hat{w}^{(t)}(\Theta)) \bigg|_{\Theta^{(t)}} = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial L_{i}^{\text{meta}}(\hat{w})}{\partial \hat{w}(\Theta)} \bigg|_{\hat{w}^{(t)}} \sum_{j=1}^{n} \frac{\partial \hat{w}^{(t)}(\Theta)}{\partial \Theta} \bigg|_{\Theta^{(t)}} \frac{\partial \phi_{j}^{(t)}}{\partial \Theta} \bigg|_{\Theta^{(t)}}.
\]

\[
= -\frac{\alpha}{n} \sum_{j=1}^{n} \left( \frac{1}{m} \sum_{i=1}^{m} G_{ij} \right) \frac{\partial \phi_{j}^{(t)}}{\partial \Theta} \bigg|_{\Theta^{(t)}},
\]

where

\[
G_{ij} = \frac{\partial L_{i}^{\text{meta}}(\hat{w})}{\partial \hat{w}} \bigg|_{\hat{w}^{(t)}}^{T} \frac{\partial L_{j}^{\text{train}}(w)}{\partial w} \bigg|_{w^{(t)}},
\]

\[
\phi_{j}^{(t)} = g_{e}(L_{j}^{\text{train}}(w^{(t)}); \Theta^{(t)}).
\]

More details on calculating Eq. (6) have been presented in supplementary material.

Some interesting observations can be attained from Eq. (4), which facilitates a better understanding to the insight why our weighting scheme can work on biased labeling samples.

By substituting Eq. (6) and (7) into Eq. (4), we can get:

\[
\Theta^{(t+1)} = \Theta^{(t)} + \frac{\alpha}{n} \sum_{j=1}^{n} G_{ij} \left( \frac{1}{m} \sum_{i=1}^{m} \nabla_{\Theta} L_{i}^{\text{meta}}(\hat{w}^{(t)}(\Theta)) \bigg|_{\Theta^{(t)}} \right) \frac{\partial \phi_{j}^{(t)}}{\partial \Theta} \bigg|_{\Theta^{(t)}}.
\]

Neglecting the coefficient \(\frac{1}{m} \sum_{i=1}^{m} G_{ij}\), it is easy to see that each term in the sum orients to the ascend gradient of the weight function \(g_{e}(L_{j}^{\text{train}}(w^{(t)}); \Theta)\), \(\frac{1}{m} \sum_{i=1}^{m} G_{ij}\), the coefficient imposed on the \(j\)-th gradient term, represents the similarity between the gradient of the \(j\)-th training sample computed on training loss and the average gradient of the mini-batch meta data calculated on meta loss. That is to say, if the learning gradient of a training sample is similar to that of the meta samples, then the sample will be considered as beneficial for getting right results and its weight tends to be more possibly increased. And conversely, the weight of the sample inclines to be suppressed.

### 3.3. The Meta-PGC Algorithm

The Meta-PGC algorithm can then be summarized in Algorithm 1. Fig. 1 illustrates its main implementation process (steps 5-7) to help readers easily understand the flow of the algorithm. All steps of the algorithm are with closed-form and explicit expressions. All computations of gradients are implemented by automatic differentiation techniques and can be generalized to any deep learning architectures for student network. The algorithm can be easily implemented using popular deep learning frameworks like PyTorch. Since both the student network and the V-Net gradually ameliorate their parameters, \(w^{(t)}\) and \(\Theta^{(t)}\), respectively, from their values calculated in the last step (as clearly shown in Fig. 1), the weights generally can be updated in a stable manner, as shown in Fig. 4.

**Algorithm 1** The Meta-PGC Algorithm

**Input:** Training data \(D\), meta-data set \(\hat{D}\), batch size \(n, m\), max iterations \(T\).

**Output:** Student network parameter \(w^{(T)}\)

1. Initialize student network parameter \(w^{(0)}\) and VNet parameter \(\Theta^{(0)}\).
2. for \(t = 0 \text{ to } T - 1\) do
3. \(\{x, y\} \leftarrow \text{SampleMiniBatch}(D, n)\).
4. \(\{x^{(\text{meta})}, y^{(\text{meta})}\} \leftarrow \text{SampleMiniBatch}(\hat{D}, m)\).
5. Formulate the student’s learning function \(w^{(t)}(\Theta)\) by Eq. (3).
6. Update \(\Theta^{(t+1)}\) by Eq. (4).
7. Update \(w^{(t+1)}\) by Eq. (5).
8. end for
The convergence of training loss can be obtained for SGD based optimization methods, as known in (Reddi et al., 2016). Here, we further prove the convergence of our method on the perspective of monotonically decreasing of the meta loss objective. The proof is listed in the supplementary material.

**Theorem 1.** Suppose the loss function $L(\cdot)$ is Lipschitz-smooth in $w$ with constant $L$, and $g_\alpha(\cdot)$ is differentiable. And the loss function $L_i(w)$ have $\sigma$-bounded gradients with respect to training/meta data. Let the learning rate $\alpha_i \leq \frac{2\sigma}{L}$, where $n$ is the training batch size. Then the meta loss in Algorithm 1 always monotonically decreases for any sequence of training mini-batches, namely,

$$\frac{1}{M} \sum_{i=1}^{M} L_{i, \text{meta}}(w^{t+1}) \leq \frac{1}{M} \sum_{i=1}^{M} L_{i, \text{meta}}(w^t).$$  (9)

### 4. Experimental Results

We evaluate the capability of our proposed algorithm and compare its performance with other state-of-the-art methods in this section.

#### 4.1. Robustness against Corrupted Labels

We first test the robustness of our algorithm on several benchmark datasets with corrupted labels with varying noise rates.

**Datasets.** Two benchmark datasets are employed: CIFAR-10 and CIFAR-100 (Krizhevsky, 2009). Both datasets are popularly used for evaluation of noisy labels in the current literatures (Ma et al., 2018; Han et al., 2018).

We study two settings of corrupted labels on training set:

- **Uniform noise.** The label of each image is independently changed to a random class with probability $p$ following the same setting in (Zhang et al., 2017).
- **Flip noise.** The label of each image is independently flipped to similar classes with total probability $p$. In our experiments, we randomly select two classes as similar classes with equal probability.

1,000 images with clean labels in validation set are randomly selected as the meta-data set for all meta-learning methods.

**Structure of student network.** We adopt a Wide ResNet-28-10 (WRN-28-10) (Zagoruyko & Komodakis, 2016) for uniform noise and ResNet32 (He et al., 2016) for flip noise as our student network models.

**Comparison methods.** We compare our algorithm with the following competing methods.

- **Reed** (Reed et al., 2015) assumes a weighted combination of predicted and original labels as the correct labels, and then does back propagation.
- **S-Model** (Goldberger & Ben-Reuven, 2017) adds an additional softmax layer after the regular classification output layer to model the noise transition matrix.
- **Self-paced** (Kumar et al., 2010) and **Focal Loss** (Lin et al., 2018) represent the state-of-the-arts of the traditional sample reweighting techniques.
- **MentorNet** (Jiang et al., 2018) designs a meta learning model to train a teacher network with a sequence losses and example weights pairs.

### Table 1. Test accuracy (%) of different models on CIFAR-10 and CIFAR-100 with varying noise rates under uniform noise.

| Datasets / Noise Rate | BaseModel | Reed-Hard | S-Model | Self-paced | Focal Loss | MentorNet | Co-teaching | D2L | Fine-tuning | L2RW | GLC | Ours |
|-----------------------|-----------|-----------|---------|------------|------------|------------|-------------|-----|-------------|------|-----|------|
| CIFAR-10 0%           | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 20%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 40%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 60%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| CIFAR-100 0%          | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 20%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 40%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 60%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |

### Table 2. Test accuracy (%) of different models on CIFAR-10 and CIFAR-100 with varying noise rates under flip noise.

| Datasets / Noise Rate | BaseModel | Reed-Hard | S-Model | Self-paced | Focal Loss | MentorNet | Co-teaching | D2L | Fine-tuning | L2RW | GLC | Ours |
|-----------------------|-----------|-----------|---------|------------|------------|------------|-------------|-----|-------------|------|-----|------|
| CIFAR-10 0%           | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 20%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 40%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 60%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| CIFAR-100 0%          | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 20%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 40%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
| 60%                   | 70.77     | 70.72     | 92.89   | 43.01      | 74.07      | 50.86      | 93.23       |     |             |      |     |      |
Figure 2. Training and test accuracy changing curves in uniform noise cases of CIFAR-10 and CIFAR-100 datasets. Solid and dotted curves denote the test and training accuracies, respectively. Our method and L2RW are less prone to overfit label noises, while our method can converge faster at around 20K steps as shown in Fig. 2(a). We thus terminate our method in 20K steps in other experiments.

- **Co-teaching** (Han et al., 2018) employs a ‘co-teaching’ fashion to train two networks simultaneously to screen small loss data.
- **D2L** (Ma et al., 2018) develops a dimensionality-driven learning strategy, which monitors the dimensionality of subspaces during training and adapts the loss function accordingly.
- **L2RW** (Ren et al., 2018) leverages an additional meta-dataset to adaptively assign weights to training examples in every iteration.
- **GLC** (Hendrycks et al., 2018) uses trusted examples in a data-efficient manner to mitigate the effects of label noise on deep CNNs.

In addition, we compare two variations of our methods for more comprehensive comparison following the setting of (Ren et al., 2018). **BaseModel** refers to the similar student network utilized in our method, while directly trained on the given training data. For a fair comparison, we conduct an additional **Fine-tuning** method, by fine-tune the result of **BaseModel** on the meta-data with clean labels to further enhance its performance. We also trained the baseline networks only on 1000 clean images, and the performance are evidently worse than proposed method due to the neglecting of the knowledge underlying large amount of training samples. We thus have not involved its results in comparison.

**Parameter Settings.** All the baseline networks were trained using SGD with momentum 0.9, weight decay $10^{-4}$ and an initial learning rate of 0.1. The learning rate is divided by 10 after 18K steps and 19K steps (for a total of 20K steps) in uniform noise, and after 20K steps and 25K steps (for a total of 30K steps) in flip noise. We repeated the experiments 5 times for CIFAR-10 and CIFAR-100 with different random seeds for network initialization and label noise generation.

**Results.** We report accuracy over 5 repetitions for each series of experiments and each competing method in Tables 1 and 2. It can be observed that our method gets the best performance across almost all datasets and all noise rates, except the second for 40% Flip noise. At 0% noise cases, our method performs only slightly worse than the BaseModel, which is directly trained on clean datasets, and should be an upper limit of our method in such cases (without corrupted labels).

Besides, it can be seen that the performance gaps between ours and all other competing methods increase as the noise
rate is increased from 40% to 60% under uniform noise. Even with 60% label noise, our method can still obtain a relatively high classification accuracy, and our method attains more than 15% accuracy gain compared with the second best result under 60% noise for CIFAR100 dataset, which indicates that our methods is potentially an effective strategy for semi/weakly-supervised learning, which will be investigated in our future research.

4.2. More Evaluations on Our Algorithm

We then depict more experimental results to evaluate more interesting capabilities of the proposed method. Since our method is inspired by L2RW (Ren et al., 2018), we compare this method, as well as the BaseModel, in this section for better illustration.

Robustness towards label noise overfitting issue. Fig. 2 plots the tendency curves of the mini-batch training accuracy calculated on noisy training set in experiments, as well as those calculated simultaneously on clean test data during learning iterations. From the figure, we can easily find that the BaseModel can easily overfit to the noisy labels contained in the training set, whose test accuracy quickly degrades after the first learning rate decays. While our method and L2RW are less prone to such overfitting issue, they retain the similar test accuracy until termination. Especially, throughout all our experiments, we find that our method can converge significantly faster than the BaseModel and L2RW methods\(^1\), as clearly shown in Fig.2, and get the peak performance at around 20K steps, as compared with 60K iterations required for the other two methods, as shown in Fig. 2(a). We thus only report our results at 20K steps in the figure for other experiments.

Understanding the weighting mechanism. It is beneficial to understand our algorithm contributes to learning more robust models during training. Through inputting training samples to the outputted student network and V-Net, we can obtain loss and corresponding weight values for all training samples. As shown in the upper right position of subplots in Fig. 3, the relation between loss and weight meets our expectation, i.e, V-Net tends to impose smaller weights to larger loss sample, which are possibly be corrupted labels and...
should be suppressed. This can be rationally explained by the fact that we should more emphasize those high-confident samples unbiased to the underlying groundtruth label distribution, while reduce the influence of those highly biased ones. Furthermore, we plot the weight distribution of clean and noisy training samples in Fig. 3. It can be seen that almost all large weights belong to clean samples, and the noisy samples’ weights all approach 0, which implies that the trained V-Net can distinguish clean and noisy images.

5. Conclusion

In this paper, we propose a novel meta-learning method for training a classifier on corrupted labels. Compared with current meta-learning methods, our method has few tuned hyper-parameters and the meta-learner’s structure is very simple. We propose an algorithm for jointly online optimizing student network and V-Net, which finely complies with a real teaching progress to a student. The work principle of our algorithm can be well explained and the procedure of our method can be easily reproduced. Our empirical results substantiate the robustness of the new algorithm on corrupted training data, as well as its stable weights learning along iterations. This weight learning mechanism is hopeful to be extended to other weight setting problems in machine learning, like ensemble methods and multi-view learning, which will be investigate in our future research.

References

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. Learning to learn by gradient descent by gradient descent. In NeurIPS, 2016.

Azadi, S., Feng, J., Jegelka, S., and Darrell, T. Auxiliary image regularization for deep cnns with noisy labels. In ICLR, 2016.

Bi, W., Wang, L., Kwok, J. T., and Tu, Z. Learning to predict from crowdsourced data. In UAI, 2014.

Chang, H.-S., Learned-Miller, E., and McCallum, A. Active bias: Training more accurate neural networks by emphasizing high variance samples. In NeurIPS, 2017.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16:321–357, 2002.

Dehghani, M., Severyn, A., Rothe, S., and Kamps, J. Learning to learn from weak supervision by full supervision. In NeurIPS Workshop, 2017.

Dehghani, M., Mehrjou, A., Gouws, S., Kamps, J., and Schölkopf, B. Fidelity-weighted learning. In ICLR, 2018.

Fan, Y., Tian, F., Qin, T., Li, X.-Y., and Liu, T.-Y. Learning to teach. 2018.

Fernando, D. I. T. and Mikhail, J. B. A framework for robust subspace learning. International Journal of Computer Vision, 54(1):117–142, 2003.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In ICML, 2017.

Franceschi, L., Frasconi, P., Salzo, S., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In ICML, 2018.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1):119–139, 1997.

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

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: robust training deep neural networks with extremely noisy labels. In NeurIPS, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In CVPR, 2016.

Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. Using trusted data to train deep networks on labels corrupted by severe noise. In NeurIPS, 2018.
Push the Student to Learn Right: Progressive Gradient Correcting by Meta-learner on Corrupted Labels

Jiang, L., Meng, D., Mitamura, T., and Hauptmann, A. G. Easy samples first: Self-paced reranking for zero-example multimedia search. In ACM MM, 2014a.

Jiang, L., Meng, D., Yu, S.-I., Lan, Z., Shan, S., and Hauptmann, A. Self-paced learning with diversity. In NeurIPS, 2014b.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In ICML, 2018.

Kawaguchi, K., Kaelbling, L. P., and Bengio, Y. Generalization in deep learning. arXiv preprint arXiv:1710.05468, 2017.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, 2009.

Kumar, M. P., Packer, B., and Koller, D. Self-paced learning for latent variable models. In NeurIPS, 2010.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. Science, 350(6266):1332–1338, 2015.

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

Liang, J., Jiang, L., Meng, D., and Hauptmann, A. Learning to detect concepts from webly-labeled video data. In IJCAI, 2016.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. IEEE transactions on pattern analysis and machine intelligence, 2018.

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S. M., Xia, S.-T., Wijewickrema, S., and Bailey, J. Dimensionality-driven learning with noisy labels. In ICML, 2018.

Malisiewicz, T., Gupta, A., and Efros, A. A. Ensemble of exemplar-svms for object detection and beyond. In ICCV, 2011.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. Learning with noisy labels. In NeurIPS, 2013.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. In NeurIPS, 2017.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. In ICLR, 2018.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In ICLR, 2017.

Reddi, S. J., Hefny, A., Sra, S., Poczos, B., and Smola, A. Stochastic variance reduction for nonconvex optimization. In ICML, 2016.

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

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

Shu, J., Xu, Z., and Meng, D. Small sample learning in big data era. arXiv preprint arXiv:1808.04572, 2018.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In NeurIPS, 2017.

Sukhbaatar, S. and Fergus, R. Learning from noisy labels with deep neural networks. In ICLR workshop, 2015.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, 40(12):3358–3378, 2007.

Vahdat, A. Toward robustness against label noise in training deep discriminative neural networks. In NeurIPS, 2017.

Van Rooyen, B. and Williamson, R. C. A theory of learning with corrupted labels. Journal of Machine Learning Research, 2018.

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., and Belongie, S. J. Learning from noisy large-scale datasets with minimal supervision. In CVPR, 2017.

Wang, Y., Kucukelbir, A., and Blei, D. M. Robust probabilistic modeling with bayesian data reweighting. In ICML, 2017.

Wu, L., Tian, F., Xia, Y., Fan, Y., Qin, T., Jian-Huang, L., and Liu, T.-Y. Learning to teach with dynamic loss functions. In NeurIPS, 2018.

Zadrozny, B. Learning and evaluating classifiers under sample selection bias. In ICML, 2004.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In BMVC, 2016.

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

Zhang, Z. and Sabuncu, M. R. Generalized cross entropy loss for training deep neural networks with noisy labels. In NeurIPS, 2018.

Zhuang, B., Liu, L., Li, Y., Shen, C., and Reid, I. D. Attend in groups: a weakly-supervised deep learning framework for learning from web data. In CVPR, 2017.