This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Learning to Learn from Noisy Labeled Data

Junnan Li[1], Yongkang Wong[1], Qi Zhao[2], Mohan S. Kankanhalli[1]
[1]National University of Singapore [2]University of Minnesota
lijunnan@u.nus.edu, yongkang.wong@nus.edu.sg, qzhao@cs.umn.edu, mohan@comp.nus.edu.sg

Abstract

Despite the success of deep neural networks (DNNs) in image classification tasks, the human-level performance relies on massive training data with high-quality manual annotations, which are expensive and time-consuming to collect. There exist many inexpensive data sources on the web, but they tend to contain inaccurate labels. Training on noisy labeled datasets causes performance degradation because DNNs can easily overfit to the label noise. To overcome this problem, we propose a noise-tolerant training algorithm, where a meta-learning update is performed prior to conventional gradient update. The proposed meta-learning method simulates actual training by generating synthetic noisy labels, and train the model such that after one gradient update using each set of synthetic noisy labels, the model does not overfit to the specific noise. We conduct extensive experiments on the noisy CIFAR-10 dataset and the Clothing1M dataset. The results demonstrate the advantageous performance of the proposed method compared to state-of-the-art baselines.

1. Introduction

One of the key reasons why deep neural networks (DNNs) have been so successful in image classification is the collections of massive labeled datasets such as COCO [14] and ImageNet [20]. However, it is time-consuming and expensive to collect such high-quality manual annotations. A single image often requires agreement from multiple annotators to reduce label error. On the other hand, there exist other less expensive sources to collect labeled data, such as search engines, social media websites, or reducing the number of annotators per image. However, those low-cost approaches introduce low-quality annotations with *label noise*. Many studies have shown that label noise can significantly affect the accuracy of the learned classifiers [2, 23, 32]. In this work, we address the following problem: how to effectively train on noisy labeled datasets?

Some methods learn with label noise by relying on human supervision to verify seed images [11, 29] or estimate label confusion [16, 31]. However, those methods exhibit a disadvantage in scalability for large datasets. On the other hand, methods without human supervision (*e.g.* label correction [18, 24] and noise correction layers [5, 23]) are scalable but less effective and more heuristic. In this work we propose a *meta-learning* based *noise-tolerant* (MLNT) training to learn from noisy labeled data without human supervision or access to any clean labels. Rather than designing a specific model, we propose a model-agnostic training algorithm, which is applicable to any model that is trained with gradient-based learning rule.

The prominent issue in training DNNs on noisy labeled data is that DNNs often overfit to the noise, which leads to performance degradation. Our method addresses this issue by optimizing for a model's parameters that are less prone to overfitting and more robust against label noise. Specifically, for each mini-batch, we propose a meta-objective to train the model, such that after the model goes through conventional gradient update, it does not overfit to the label noise. The proposed meta-objective encourages the model to produce consistent predictions after it is trained on a variety of synthetic noisy labels. The key idea of our method is: *a noise-tolerant model should be able to consistently learn the underlying knowledge from data despite different label noise*. The main contribution of this work are as follows.

- We propose a noise-tolerant training algorithm, where a meta-objective is optimized before conventional training. Our method can be theoretically applied to any model trained with gradient-based rule. We aim to optimize for a model that does not overfit to a wide spectrum of artificially generated label noise.
- We formulate our meta-objective as: train the model such that after it learns from various synthetic noisy labels using gradient update, the updated models give consistent predictions with a teacher model. We adapt a self-ensembling method to construct the teacher model, which gives more reliable predictions unaffected by the synthetic noise.
- We perform experiments on two datasets with synthetic and real-world label noise, and demonstrate the

advantageous performance of the proposed method in image classification tasks compared to state-of-the-art methods. In addition, we conduct extensive ablation study to examine different components of the proposed method. Our code is publicly available[1].

2. Related Work

Learning with label noise. A number of approaches have been proposed to train DNNs with noisy labeled data. One line of approaches formulate explicit or implicit noise models to characterize the distribution of noisy and true labels, using neural networks [5, 8, 11, 19, 16, 23, 29], directed graphical models [31], knowledge graphs [13], or conditional random fields [27]. The noise models are then used to infer the true labels or assign smaller weights to noisy samples. However, these methods often require a small set of data with clean labels to be available, or use expensive estimation methods. They also rely on specific assumptions about the noise model, which may limit their effectiveness with complicated label noise. Another line of approaches use correction methods to reduce the influence of noisy labels. Bootstrap method [18] introduces a consistency objective that effectively re-labels the data during training. Tanaka *et al*. [24] propose to jointly optimize network parameters and data labels. An iterative training method is proposed to identify and downweight noisy samples [30]. A few other methods have also been proposed that use noise-tolerant loss functions to achieve robust learning under label noise [3, 4, 28].

Meta-Learning. Recently, meta-learning methods for DNNs have resurged in its popularity. Meta-learning generally seeks to perform the learning at a level higher than where conventional learning occurs, *e.g*. learning the update rule of a learner [17], or finding weight initializations that can be easily fine-tuned [1] or transferred [12]. Our approach is most related to MAML [1], which aims to train model parameters that can learn well based on a few examples and a few gradient descent steps. Both MAML and our method are model-agnostic and perform training by doing gradient updates on simulated meta-tasks. However, our objective and algorithm are different from that of MAML. MAML addresses few-shot transfer to new tasks, whereas we aim to learn a noise-tolerant model. Moreover, MAML trains using classification loss on a meta-test set, whereas we use a consistency loss with a teacher model.

Self-Ensembling. Several recent methods based on self-ensembling have improved the state-of-the-art results for semi-supervised learning [10, 21, 25], where labeled samples are scarce and unlabeled samples are abundant. These methods apply a consistency loss to the unlabeled samples, which regularizes a neural network to make consistent predictions for the same samples under different data augmentation [21], dropout and noise conditions [10]. We focus in particular on the self-ensembling approach proposed by Tarvainen & Valpola [25] as it forms one of the basis of our approach. Their approach proposes two networks: a *student* network and a *teacher* network, where the weights of the teacher are the exponential moving average of those of the student. They enforce the student network to make consistent predictions with the teacher network. In our method, we use the teacher network in meta-test to train the student network such that it is more tolerant to label noise.

[1] https://github.com/LiJunnan1992/MLNT

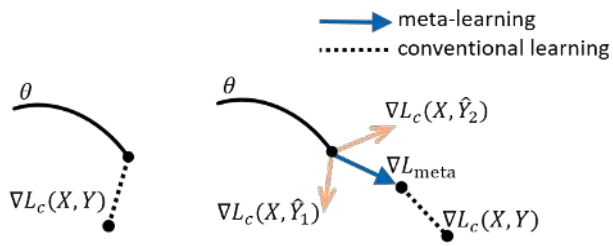


Figure 1: Left: conventional gradient update with cross entropy loss may overfit to label noise. Right: a meta-learning update is performed beforehand using synthetic label noise, which encourages the network parameters to be noise-tolerant and reduces overfitting during the conventional update.

3. Method

3.1. Problem Statement

We consider a classification problem with a training set $\mathcal{D} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), ..., (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$, where $\boldsymbol{x}_i$ denotes the ith sample and $\boldsymbol{y}_i \in \{0, 1\}^c$ is a one-hot vector representing the corresponding noisy label over c classes. Let $f(\boldsymbol{x}_i, \boldsymbol{\theta})$ denotes the discriminative function of a neural network parameterized by $\boldsymbol{\theta}$, which maps an input to an output of the c-class softmax layer. The conventional objective for supervised classification is to minimize an empirical risk, such as the cross entropy loss:

$$\mathcal{L}_c = -\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{y}_i \cdot \log(f(\boldsymbol{x}_i, \boldsymbol{\theta})), \tag{1}$$

where $\cdot$ denotes dot product.

However, since $\boldsymbol{y}_i$ contains noise, the neural network can overfit and perform poorly on the test set. We propose a meta-objective that encourages the network to learn noise-tolerant parameters. The details are delineated next.

3.2. Meta-Learning based Noise-Tolerant Training

Our method can learn the parameters of a DNN model in such a way as to "prepare" the model for label noise. The intuition behind our method is that when training with a

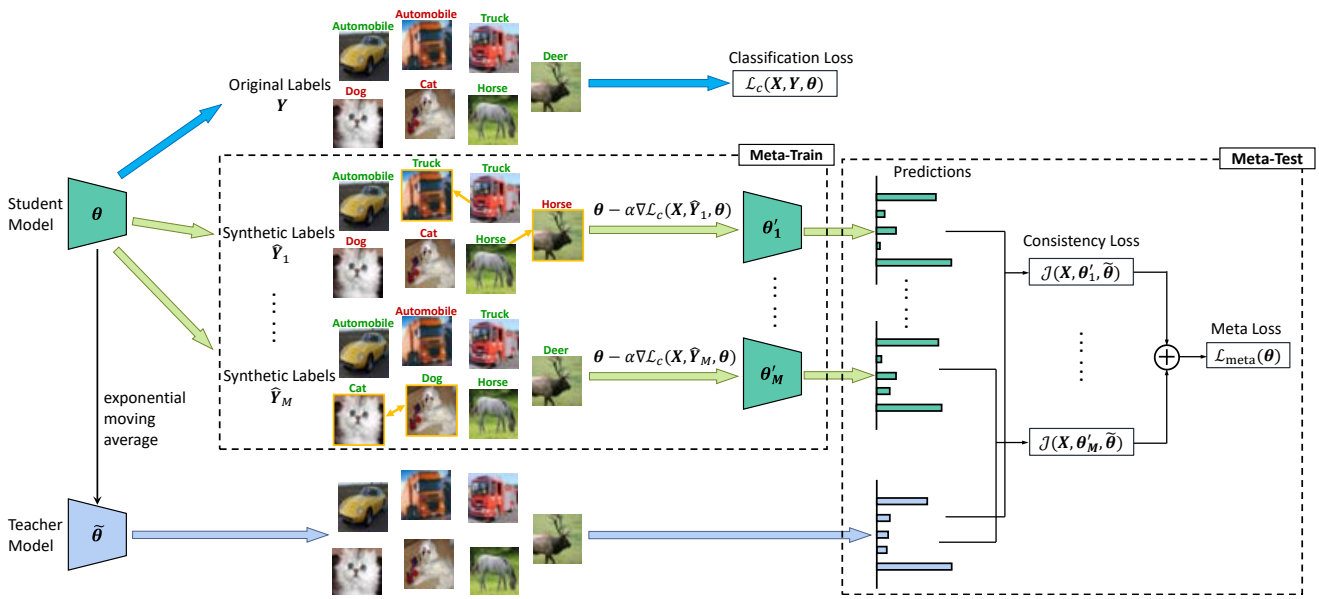


Figure 2: Illustration of the proposed meta-learning based noise-tolerant (MLNT) training. For each mini-batch of training data, a meta loss is minimized before training on the conventional classification loss. We first generate multiple mini-batches of synthetic noisy labels with random neighbor label transfer (marked by orange arrow). The random neighbor label transfer can preserve the underlying noise transition (*e.g.* DEER $\rightarrow$ HORSE, CAT $\leftrightarrow$ DOG), therefore generating synthetic label noise in a similar distribution as the original data. For each synthetic mini-batch, we update the parameters with gradient descent, and enforce the updated model to give consistent predictions with a *teacher* model. The meta-objective is to minimize the consistency loss across all updated models *w.r.t* $\boldsymbol{\theta}$.

gradient-based rule, some network parameters are more tolerant to label noise than others. How can we encourage the emergence of such noise-tolerant parameters? We achieve this by introducing a meta-learning update before the conventional update for each mini-batch. The meta-learning update simulates the process of training with label noise and makes the network less prone to over-fitting. Specifically, for each mini-batch of training data, we generate a variety of synthetic noisy labels on the same images. With each set of synthetic noisy labels, we update the network parameters using one gradient update, and enforce the updated network to give consistent predictions with a *teacher* model unaffected by the synthetic noise. As shown in Figure 1, the meta-learning update optimizes the model so that it can learn better with conventional gradient update on the original mini-batch. In effect, we aim to find model parameters that are less sensitive to label noise and can consistently learn the underlying knowledge from data despite label noise. The proposed meta-learning update consists of two procedures: meta-train and meta-test.

Meta-Train. Formally, at each training step, we consider a mini-batch of data $(\boldsymbol{X}, \boldsymbol{Y})$ sampled from the training set, where $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_k\}$ are k samples, and $\boldsymbol{Y} = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_k\}$ are the corresponding noisy labels. We want to generate multiple mini-batches of noisy labels $\{\hat{\boldsymbol{Y}}_1, ..., \hat{\boldsymbol{Y}}_M\}$ with similar label noise distribution as $\boldsymbol{Y}$. We will describe the procedure for generating one set of noisy labels $\hat{\boldsymbol{Y}}_m = \{\hat{\boldsymbol{y}}_1^m, ..., \hat{\boldsymbol{y}}_k^m\}$. First, we randomly select ρ samples out of the mini-batch of k samples. For each selected sample $\boldsymbol{x}_i$, we rank its neighbors within the mini-batch. Then we randomly select a neighbor $\boldsymbol{x}_j$ from its top 10 nearest neighbors (10 is experimentally determined), and use the neighbor's label $\boldsymbol{y}_j$ to replace the label for $\boldsymbol{x}_i$, $\hat{\boldsymbol{y}}_i^m = \boldsymbol{y}_j$. Because we transfer labels among neighbors, the synthetic noisy labels are from a similar distribution as the original noisy labels. We repeat the above procedure M times to generate M mini-batches of synthetic noisy labels. Note that we compute nearest neighbors based on the Euclidean distance between feature representations (pre-softmax layer activations) generated by a DNN pre-trained on the entire noisy training set $\mathcal{D}$.

Let $\boldsymbol{\theta}$ denote the current model's parameters, for each synthetic mini-batch $(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m)$, we update $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'_m$ using one gradient descent step on the mini-batch.

$$\boldsymbol{\theta}'_m = \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta}), \tag{2}$$

where $\mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta})$ is the cross entropy loss described in equation 1, and α is the step size.

Meta-Test. Our meta-objective is to train $\boldsymbol{\theta}$ such that each updated parameters $\boldsymbol{\theta}'_m$ do not overfit to the specific noisy labels $\hat{\boldsymbol{Y}}_m$. We achieve this by enforcing each updated model to give consistent predictions with a *teacher* model. We consider the model parameterized by $\boldsymbol{\theta}$ as the

student model, and construct the *teacher* model parameterized by $\tilde{\boldsymbol{\theta}}$ following the self-ensembling [25] method. The teacher model usually gives better predictions than the student model. Its parameters are computed as the exponential moving average (EMA) of the student's parameters. Specifically, at each training step, we update $\tilde{\boldsymbol{\theta}}$ with:

$$\tilde{\boldsymbol{\theta}} = \gamma\tilde{\boldsymbol{\theta}} + (1-\gamma)\boldsymbol{\theta}, \tag{3}$$

where γ is a smoothing coefficient hyper-parameter.

Since the teacher model is unaffected by the synthetic label noise, we enforce a consistency loss $\mathcal{J}(\boldsymbol{\theta}'_m)$ that encourages each updated model (with parameters $\boldsymbol{\theta}'_m$) to give consistent predictions with the teacher model on the same input $\boldsymbol{X}$. We define $\mathcal{J}(\boldsymbol{\theta}'_m)$ as the Kullback-Leibler (KL)-divergence between the softmax predictions from the updated model $f(\boldsymbol{X}, \boldsymbol{\theta}'_m)$ and the softmax predictions from the teacher model $f(\boldsymbol{X}, \tilde{\boldsymbol{\theta}})$. We find that KL-divergence produces better results compared to the mean squared error used in [25].

$$\mathcal{J}(\boldsymbol{\theta}'_m) = \frac{1}{k}\sum_{i=1}^{k} D_{KL}(f(\boldsymbol{x}_i, \tilde{\boldsymbol{\theta}})||f(\boldsymbol{x}_i, \boldsymbol{\theta}'_m)) \tag{4}$$

$$= \frac{1}{k}\sum_{i=1}^{k} \mathbb{E}(\log(f(\boldsymbol{x}_i, \tilde{\boldsymbol{\theta}})) - \log(f(\boldsymbol{x}_i, \boldsymbol{\theta}'_m))). \tag{5}$$

We want to minimize the consistency loss for all of the M updated models with parameters $\{\boldsymbol{\theta}'_1, ..., \boldsymbol{\theta}'_M\}$. Therefore, the meta loss is defined as the average of all consistency losses:

$$\mathcal{L}_{\text{meta}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{m=1}^{M} \mathcal{J}(\boldsymbol{\theta}'_m) \tag{6}$$

$$= \frac{1}{M}\sum_{m=1}^{M} \mathcal{J}(\boldsymbol{\theta} - \alpha\nabla_\theta \mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta})). \tag{7}$$

Although $\mathcal{L}_{\text{meta}}(\boldsymbol{\theta})$ is computed using the updated model parameters $\boldsymbol{\theta}'_m$, the optimization is performed over the student model parameters $\boldsymbol{\theta}$. We perform stochastic gradient descent (SGD) to minimize the meta loss. The model parameters $\boldsymbol{\theta}$ are updated as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla\mathcal{L}_{\text{meta}}(\boldsymbol{\theta}), \tag{8}$$

where η is the meta-learning rate.

After the meta-learning update, we perform SGD to optimize the classification loss on the original mini-batch $(\boldsymbol{X}, \boldsymbol{Y})$.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta\nabla\mathcal{L}_c(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}), \tag{9}$$

where β is the learning rate. The full algorithm is outlined in Algorithm 1.

Algorithm 1 Meta-Learning based Noise-Tolerant Training

Randomly initialize $\boldsymbol{\theta}$
Initialize teacher model $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}$
while not done **do**
  Sample a mini-batch $(\boldsymbol{X}, \boldsymbol{Y})$ of size k from $\mathcal{D}$.
  for $m = 1 : M$ **do**
    Generate synthetic noisy labels $\hat{\boldsymbol{Y}}_m$ by random neighbor label transfer
    Compute updated parameters with gradient descent: $\boldsymbol{\theta}'_m = \boldsymbol{\theta} - \alpha\nabla_\theta\mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta})$
    Evaluate consistency loss with teacher: $\mathcal{J}(\boldsymbol{\theta}'_m) = \frac{1}{k}\sum_{i=1}^{k} D_{KL}(f(\boldsymbol{x}_i, \tilde{\boldsymbol{\theta}})||f(\boldsymbol{x}_i, \boldsymbol{\theta}'_m))$
  end for
  Evaluate $\mathcal{L}_{\text{meta}}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{m=1}^{M} \mathcal{J}(\boldsymbol{\theta}'_m)$
  Meta-learning update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla\mathcal{L}_{\text{meta}}(\boldsymbol{\theta})$
  Evaluate classification loss $\mathcal{L}_c(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$
  Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta\nabla\mathcal{L}_c(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$
  Update teacher model: $\tilde{\boldsymbol{\theta}} = \gamma\tilde{\boldsymbol{\theta}} + (1-\gamma)\boldsymbol{\theta}$
end while

Note that the meta-gradient $\nabla\mathcal{L}_{\text{meta}}(\boldsymbol{\theta})$ involves a gradient through a gradient, which requires calculating the second-order derivatives with respect to $\boldsymbol{\theta}$. In our experiments we use a first-order approximation by omitting the second-order derivatives, which can significantly increase the computation speed. The comparison in Section 4.4 shows that this approximation performs almost as well as using second-order derivatives. This provides another intuition to explain our method: The first-order approximation considers the term $\alpha\nabla_\theta\mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta})$ in equation 7 as a constant. Therefore, we can consider the update $\boldsymbol{\theta} - \alpha\nabla_\theta\mathcal{L}_c(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m, \boldsymbol{\theta})$ as injecting data-dependent noise to the parameters, and adding noise to the network during training has been shown by many studies to have a regularization effect [10, 22].

3.3. Iterative Training

We propose an iterative training scheme for two purposes: (1) Remove samples with potentially wrong class labels from the classification loss $\mathcal{L}_c(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$. (2) Improve the predictions from the teacher model $f(\boldsymbol{x}_i, \tilde{\boldsymbol{\theta}})$ so that the consistency loss is more effective.

First, we perform an initial training iteration following the method described in Algorithm 1, and acquire a model with the best validation accuracy (usually the teacher). We name that model as *mentor* and use $\boldsymbol{\theta}^*$ to denote its parameters. In the second training iteration, we repeat the steps in Algorithm 1 with two changes described as follows.

First, if the classification loss $\mathcal{L}_c(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$ is applied to the entire training set $\mathcal{D}$, samples with wrong ground-truth labels can corrupt training. Therefore, we remove a sam-

ple from the classification loss if the mentor model assigns a low probability to the ground-truth class. In effect, the classification loss would now sample batches from a filtered training set $\mathcal{D}'$ which contains fewer corrupted samples.

$$\mathcal{D}' = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D} \mid \boldsymbol{y}_i \cdot f(\boldsymbol{x}_i, \boldsymbol{\theta}^*) > \tau\}, \tag{10}$$

where $f(\boldsymbol{x}_i, \boldsymbol{\theta}^*)$ is the softmax prediction of the mentor model, and τ is a threshold to control the balance between the quality and quantity of $\mathcal{D}'$.

Second, we improve the effectiveness of the consistency loss by merging the predictions from the mentor model and the teacher model to produce more reliable predictions. The new consistency loss is:

$$\mathcal{J}'(\boldsymbol{\theta}'_m) = \frac{1}{k} \sum_{i=1}^{k} D_{KL}(\boldsymbol{s} || f(\boldsymbol{x}_i, \boldsymbol{\theta}'_m)), \tag{11}$$

$$\boldsymbol{s} = \lambda f(\boldsymbol{x}_i, \tilde{\boldsymbol{\theta}}) + (1-\lambda) f(\boldsymbol{x}_i, \boldsymbol{\theta}^*)), \tag{12}$$

where λ is a weight to control the importance of the teacher model and the mentor model. It is ramped up from 0 to 0.5 as training proceeds.

We train for three iterations in our experiments. The mentor model is the best model from the previous iteration. We observe that further training iterations beyond three do not give noticeable performance improvement.

4. Experiments

4.1. Datasets

We conduct experiments on two datasets, namely CIFAR-10 [9] and Clothing1M [31]. We follow the same experimental setting as previous studies [16, 24, 27] for fair comparison.

For CIFAR-10, we split 10% of the training data for validation, and artificially corrupt the rest of the training data with two types of label noise: symmetric and asymmetric. The symmetric label noise is injected by using a random one-hot vector to replace the ground-truth label of a sample with a probability of r. The asymmetric label noise is designed to mimic some of the structure of real mistakes for similar classes [16]: TRUCK $\rightarrow$ AUTOMOBILE, BIRD $\rightarrow$ AIRPLANE, DEER $\rightarrow$ HORSE, CAT $\leftrightarrow$ DOG. Label transitions are parameterized by $r \in [0, 1]$ such that true class and wrong class have probability of $1 - r$ and r, respectively.

Clothing1M [31] consists of 1M images collected from online shopping websites, which are classified into 14 classes, *e.g.* t-shirt, sweater, jacket. The labels are generated using surrounding texts provided by sellers, which contain real-world errors. We use the $14k$ and $10k$ clean data for validation and test, respectively, but we do not use the $50k$ clean training data.

4.2. Implementation

For experiments on CIFAR-10, we follow the same experimental setting as [24] and use the network based on Pre-Act ResNet-32 [7]. By common practice [24], we normalize the images, and perform data augmentation by random horizontal flip and 32×32 random cropping after padding 4 pixels per side. We use a batch size $k = 128$, a step size $\alpha = 0.2$, a learning rate $\beta = 0.2$, and update $\boldsymbol{\theta}$ using SGD with a momentum of 0.9 and a weight decay of 10^{-4}. For each training iteration, we divide the learning rate by 10 after 80 epochs, and train until 120 epochs. For the initial iteration, we ramp up η (meta-learning rate) from 0 to 0.4 during the first 20 epochs, and keep $\eta = 0.4$ for the rest of the training. In terms of the EMA decay γ, we use $\gamma = 0.99$ for the first 20 epochs and $\gamma = 0.999$ later on, because the student improves quickly early in the training, and thus the teacher should have a shorter memory [25]. In the ablation study (Section 4.4), we will show the effect of the three important hyper-parameters, namely M, the number of synthetic mini-batches, ρ, the number of samples with label replacement, and the threshold τ for data filtering. The value for all hyper-parameters are determined via validation.

For experiments on Clothing1M, we follow previous works [16, 24] and use the ResNet-50 [6] pre-trained on ImageNet. For preprocessing, we resize the image to 256×256, crop the middle 224×224 as input, and perform normalization. We use a batch size $k = 32$, a step size $\alpha = 0.02$, a learning rate $\beta = 0.0008$, and update $\boldsymbol{\theta}$ using SGD with a momentum of 0.9 and a weight decay of 10^{-3}. We train for 3 epochs for each iteration. During the first 2000 mini-batches in the initial iteration, we ramp up η from 0 to 0.0008, and set $\gamma = 0.99$. For the rest of the training, we use $\eta = 0.0008$ and $\gamma = 0.999$. Other hyper-parameters are set as $M = 10$, $\rho = 0.5k$, and $\tau = 0.3$.

4.3. Experiments on CIFAR-10

We compare the proposed MLNT with multiple baseline methods using CIFAR-10 dataset with symmetric label noise (noise ratio $r = 0, 0.1, 0.3, 0.5, 0.7, 0.9$) and asymmetric label noise (noise ratio $r = 0.1, 0.2, 0.3, 0.4, 0.5$). The baselines include:

(1) **Cross Entropy**: conventional training without the meta-learning update. We report both the results from [24] and from our own implementation.

(2) **Forward** [16]: forward loss correction based on the noise transition matrix.

(3) **CNN-CRF** [27]: a CRF model is proposed to represent the relationship between noisy and clean labels. It requires a small set of clean labels during training.

(4) **Joint Optimization** [24]: alternatively updates network parameters and corrects labels during training.

Both Forward [16] and CNN-CRF [27] require the ground-truth noise transition matrix, and Joint Optimiza-

Table 1: Classification accuracy (%) on CIFAR-10 test set for different methods trained with *symmetric* label noise. We report the mean and standard error across 5 runs.

Method	$r = 0$	$r = 0.1$	$r = 0.3$	$r = 0.5$	$r = 0.7$	$r = 0.9$
Cross Entropy [24]	93.5	91.0	88.4	85.0	78.4	41.1
Cross Entropy (reproduced)	91.84±0.05	90.33±0.06	87.85±0.08	84.62±0.08	78.06±0.16	45.85±0.91
Joint Optimization [24]	93.4	92.7	91.4	89.6	85.9	58.0
MLNT-student (1st iter.)	93.18±0.07	92.16±0.05	90.57±0.08	87.68±0.06	81.96±0.19	55.45±1.11
MLNT-teacher (1st iter.)	93.21±0.07	92.43±0.05	91.06±0.07	88.43±0.05	83.27±0.22	57.39±1.13
MLNT-student (2nd iter.)	93.24±0.09	92.63±0.07	91.99±0.13	89.71±0.07	86.28±0.19	58.21±1.09
MLNT-teacher (2nd iter.)	93.35±0.07	92.91±0.09	91.89±0.06	90.03±0.08	86.24±0.18	58.33±1.10
MLNT-student (3nd iter.)	93.29±0.08	92.91±0.10	92.02±0.09	90.27±0.10	86.95±0.17	58.57±1.12
MLNT-teacher (3nd iter.)	**93.52**±0.08	**93.24**±0.08	**92.50**±0.07	**90.65**±0.09	**87.11**±0.19	**59.09**±1.12

Table 2: Classification accuracy (%) on CIFAR-10 test set for different methods trained with *asymmetric* label noise. We report the mean and standard error across 5 runs.

Method	$r = 0.1$	$r = 0.2$	$r = 0.3$	$r = 0.4$	$r = 0.5$
Cross Entropy [24]	91.8	90.8	90.0	87.1	77.3
Cross Entropy (reproduced)	91.04±0.07	90.19±0.09	88.88±0.06	86.34±0.22	77.48±0.79
Forward [16]	92.4	91.4	91.0	90.3	83.8
CNN-CRF [27]	92.0	91.5	90.7	89.5	84.0
Joint Optimization [24]	93.2	92.7	92.4	91.5	**84.6**
MLNT-student (1st iter.)	92.89±0.11	91.84±0.10	90.55±0.09	88.70±0.13	79.95±0.71
MLNT-teacher (1st iter.)	93.05±0.10	92.19±0.09	91.47±0.04	88.69±0.08	78.44±0.45
MLNT-student (2nd iter.)	93.01±0.12	92.65±0.11	91.87±0.12	90.60±0.12	81.53±0.66
MLNT-teacher (2nd iter.)	93.33±0.13	92.97±0.11	92.43±0.19	90.93±0.15	81.47±0.54
MLNT-student (3nd iter.)	93.36±0.14	92.98±0.13	92.59±0.10	91.87±0.12	82.25±0.68
MLNT-teacher (3nd iter.)	**93.61**±0.10	**93.25**±0.12	**92.82**±0.18	**92.30**±0.10	82.09±0.47

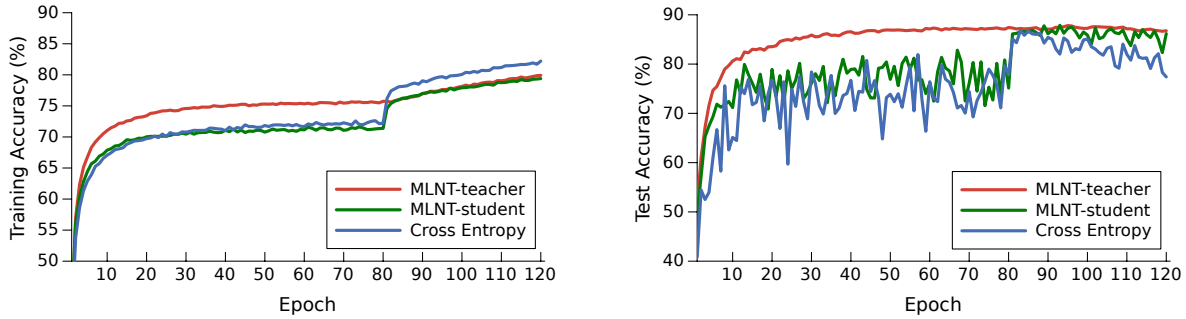
tion [24] requires the ground-truth class distribution among training data. Our method does not require any prior knowledge on the data, thus is more general. Note that all baselines use the same network architecture as our method. We report the numbers published in [24].

Table 1 and Table 2 show the results for symmetric and asymmetric label noise, respectively. Our implementation of Cross Entropy has lower overall accuracy compared to [24]. The reason could be the different programming frameworks used (we use PyTorch [15], whereas [24] used Chainer [26]). For both types of noise, the proposed MLNT method with one training iteration significantly improves accuracy compared to Cross Entropy (reproduced), and achieves comparable performance to state-of-the-art methods. Iterative training further improves the performance. MLNT-teacher after three iterations significantly outperforms previous methods. An exception where MLNT does not outperform baselines is with 50% asymmetric label noise. This is because that asymmetric label noise is generated by exchanging CAT and DOG classes, and it is theoretically impossible to distinguish them without prior knowledge when the noise ratio is 50%.

In Table 1, we also show the results with clean training data ($r = 0$). The proposed MLNT can achieve an improvement of $+1.68$ in accuracy compared to Cross Entropy, which shows the regularization effect of the proposed meta-objective.

4.4. Ablation Study

Progressive Comparison. Figure 3 plots the model's accuracy on noisy training data and its test accuracy on clean test data as training proceeds. We show a representative training process using asymmetric label noise with $r = 0.4$. Accuracy is calculated every epoch, and training accuracy is computed across all mini-batches within the epoch. Comparing the proposed MLNT methods (1st iter.) with Cross Entropy, MLNT learns more quickly during the beginning of training, as shown by the higher test accuracy. Cross Entropy has the most unstable training process, as shown by its fluctuating test accuracy curve. MLNT-student is more stable because of the regularization from the meta-objective, whereas MLNT-teacher is extremely stable because its parameters change smoothly during training. At the 80th epoch, the learning rate is divided by 10, which



(a) Training accuracy on noisy training data

(b) Test accuracy on clean test data

Figure 3: Progressive performance comparison of the proposed MLNT and Cross Entropy as training proceeds.

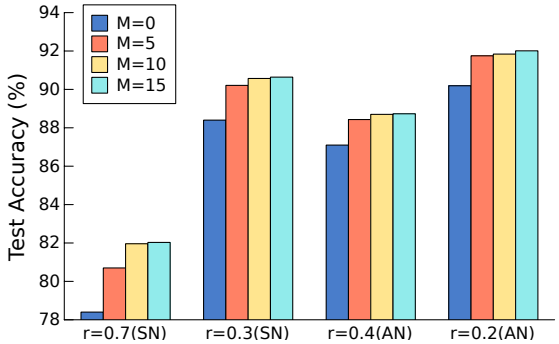


Figure 4: Performance of MLNT-student (1st iter.) on CIFAR-10 trained with different number of synthetic mini-batches M.

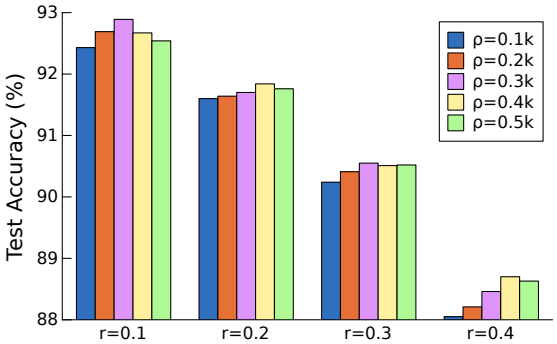


Figure 5: Performance of MLNT-student (1st iter.) on CIFAR-10 trained with asymmetric label noise using different ρ.

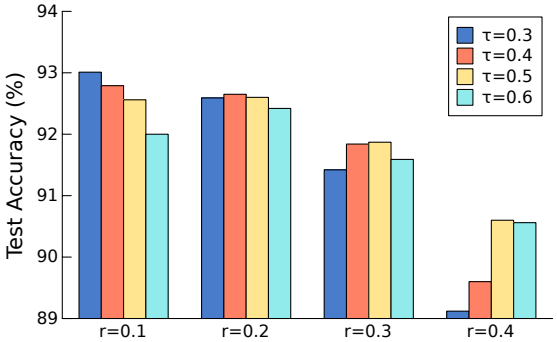


Figure 6: Performance of MLNT-student (2nd iter.) on CIFAR-10 trained with asymmetric label noise using different τ.

causes a drastic increase in both training and test accuracy for MLNT-student and Cross Entropy. After the 80th epoch, the model begins to overfit because of the small learning rate. However, the proposed MLNT-student suffers less overfitting compared to Cross Entropy, as shown by its lower training accuracy and higher test accuracy.

Hyper-parameters. We conduct ablation study to examine the effect of three hyper-parameters: M, ρ, τ. M is the number of mini-batches $\{(\boldsymbol{X}, \hat{\boldsymbol{Y}}_m)\}_{m=1}^{M}$ with synthetic noisy labels that we generate for each mini-batch $(\boldsymbol{X}, \boldsymbol{Y})$ from the original training data. Intuitively, with larger M, the model is exposed to a wider variety of label noise, and thus can learn to be more noise-tolerant. In Figure 4 we show the test accuracy on CIFAR-10 for MLNT-student (1st iter.) with $M = 0, 5, 10, 15$ ($M = 0$ is the same as Cross Entropy) trained using labels with symmetric noise (SN) and asymmetric noise (AN) of different ratio. The result shows that the accuracy indeed increases as M increases. The increase is most significant when M changes from 0 to 5, and is marginal when M changes from 10 to 15. Therefore, the experiments in this paper are conducted using $M = 10$, as a trade-off between the training speed and the model's performance.

ρ is the number of samples whose labels are changed in each synthetic mini-batch $\hat{\boldsymbol{Y}}_m$ of size k. We experiment with $\rho = 0.1k, 0.2k, 0.3k, 0.4k, 0.5k$, which correspond to $13, 26, 39, 51, 64$ samples with a batch size of 128. Figure 5 shows the performance of MLNT-student (1st iter.) using different ρ trained on CIFAR-10 with different ratio of asymmetric label noise. The performance is insensitive to the value of ρ. For different noise ratio, the optimal ρ generally falls into the range of $[0.3k, 0.5k]$.

τ is the threshold to determine which samples are filtered out by the mentor model during the 2nd and 3nd training iteration. It controls the balance between the quality and quantity of the data that is used by the classification loss. In

Figure 7: Example images from Clothing1M that are filtered out by the mentor model. We show the ground-truth label (red) and the label predicted by the mentor (blue) with their corresponding probability scores.

Figure 6 we show the performance of MLNT-student (2nd iter.) trained using different value of τ. As the noise ratio r increases, the optimal value of τ also increases to filter out more samples. Figure 7 shows some example images from Clothing1M dataset that are filtered out and their corresponding probability scores given by the mentor model.

Full optimization. We have been using a first-order approximation to optimize $\mathcal{L}_{\text{meta}}$ for faster computation speed. Here we conduct experiments using full optimization by including the second-order derivative with respect to $\boldsymbol{\theta}$. Table 3 shows the comparison on four representative sets of experiments with different label noise. We show the test accuracy (averaged across 5 runs) of MLNT-student (1st iter.) trained with full optimization and first-order approximation. The result shows that the performance from first-order approximation is nearly the same as that obtained with full second derivatives. This suggests that the improvement of MLNT mostly comes from the gradients of the meta loss at the updated parameter values, rather than the second-order gradients.

4.5. Experiments on Clothing1M

We demonstrate the efficacy of the proposed method on real-world noisy labels using the Clothing1M dataset. The results are shown in Table 4. We show the accuracy for baselines #1 and #3 reported in [24], and the accuracy for #2 reported in [16]. The proposed MLNT method with one training iteration achieves better performance compared to state-of-the-art methods. After three training iterations, MLNT achieves a significant improvement in accuracy of $+4.19$ over Cross Entropy, and an improvement of $+1.31$ over the best baseline method #3.

Table 3: Test accuracy (%) on CIFAR-10 for MNLT-student (1st iter.) with full optimization of the meta-loss and its first-order approximation.

Optimization	SN		AN	
	$r = 0.3$	$r = 0.7$	$r = 0.2$	$r = 0.4$
First-order approx.	90.57	81.96	91.84	88.70
Full	90.74	82.05	91.89	88.91

Table 4: Classification accuracy (%) of different methods on the Clothing1M test set.

Method	Accuracy
#1 Cross Entropy [24]	69.15
Cross Entropy (reproduced)	69.28
#2 Forward [16]	69.84
#3 Joint Optimization [24]	72.16
MLNT-student (1st iter.)	72.34
MLNT-teacher (1st iter.)	72.08
MLNT-student (2nd iter.)	73.13
MLNT-teacher (2nd iter.)	73.10
MLNT-student (3nd iter.)	73.44
MLNT-teacher (3nd iter.)	**73.47**

5. Conclusion

In this paper, we propose a meta-learning method to learn from noisy labeled data, where a meta-learning update is performed prior to conventional gradient update. The proposed meta-objective aims to find noise-tolerant model parameters that are less prone to overfitting. In the meta-train step, we generate multiple mini-batches with synthetic noisy labels, and use them to update the parameters. In the meta-test step, we apply a consistency loss between each updated model and a teacher model, and train the original parameters to minimize the total consistency loss. In addition, we propose an iterative training scheme, where the model from previous iteration is used to clean data and refine predictions. We evaluate the proposed method on two datasets. The results validate the advantageous performance of our method compared to state-of-the-art methods. For future work, we plan to explore using the proposed model-agnostic method to other domains with different model architectures, such as learning Recurrent Neural Networks for machine translation with corrupted ground-truth sentences.

Acknowledgment

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Strategic Capability Research Centres Funding Initiative.

References

[1] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135, 2017. 2

[2] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2014. 1

[3] Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks. In *AAAI*, pages 1919–1925, 2017. 2

[4] Aritra Ghosh, Naresh Manwani, and P. S. Sastry. Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107, 2015. 2

[5] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *ICLR*, 2017. 1, 2

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, volume 9908 of *Lecture Notes in Computer Science*, pages 630–645, 2016. 5

[8] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, pages 2309–2318, 2018. 2

[9] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Mater's thesis, University of Toronto, 2009. 5

[10] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 2, 4

[11] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *CVPR*, pages 5447–5456, 2018. 1, 2

[12] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018. 2

[13] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *ICCV*, pages 1928–1936, 2017. 2

[14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755, 2014. 1

[15] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS Workshop*, 2017. 6

[16] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pages 2233–2241, 2017. 1, 2, 5, 6, 8

[17] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 2

[18] Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, 2015. 1, 2

[19] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, pages 4331–4340, 2018. 2

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1

[21] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, pages 1163–1171, 2016. 2

[22] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 4

[23] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. In *ICLR Workshop*, 2015. 1, 2

[24] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, pages 5552–5560, 2018. 1, 2, 5, 6, 8

[25] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, pages 1195–1204, 2017. 2, 4, 5

[26] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. Chainer: a next-generation open source framework for deep learning. In *NIPS Workshop*, 2015. 6

[27] Arash Vahdat. Toward robustness against label noise in training deep discriminative neural networks. In *NIPS*, pages 5601–5610, 2017. 2, 5, 6

[28] Brendan van Rooyen, Aditya Krishna Menon, and Robert C. Williamson. Learning with symmetric label noise: The importance of being unhinged. In *NIPS*, pages 10–18, 2015. 2

[29] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge J. Belongie. Learning from noisy large-scale datasets with minimal supervision. In *CVPR*, pages 6575–6583, 2017. 1, 2

[30] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels. In *CVPR*, pages 8688–8696, 2018. 2

[31] 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. 1, 2, 5

[32] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017. 1