MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels

Lu Jiang [1] **Zhengyuan Zhou** [2] **Thomas Leung** [1] **Li-Jia Li** [1] **Li Fei-Fei** [1 2]

Abstract

Recent deep networks are capable of memorizing the entire data even when the labels are completely random. To overcome the overfitting on corrupted labels, we propose a novel technique of learning another neural network, called MentorNet, to supervise the training of the base deep networks, namely, StudentNet. During training, MentorNet provides a curriculum (sample weighting scheme) for StudentNet to focus on the sample the label of which is probably correct. Unlike the existing curriculum that is usually predefined by human experts, MentorNet learns a data-driven curriculum dynamically with StudentNet. Experimental results demonstrate that our approach can significantly improve the generalization performance of deep networks trained on corrupted training data. Notably, to the best of our knowledge, we achieve the best-published result on WebVision, a large benchmark containing 2.2 million images of real-world noisy labels. The code are at https://github.com/google/mentornet.

1. Introduction

Zhang *et al.* (2017a) found that deep convolutional neural networks (CNNs) are capable of memorizing the entire data even with corrupted labels, where some or all true labels are replaced with random labels. It is a consensus that deeper CNNs usually lead to better performance. However, the ability of deep CNNs to overfit or memorize the corrupted labels can lead to very poor generalization performance (Zhang et al., 2017a). Recently, Neyshabur *et al.* (2017) and Arpit *et al.* (2017) proposed deep learning generalization theories to explain this interesting phenomenon.

This paper studies how to overcome the corrupted label for deep CNNs, so as to improve generalization performance on the clean test data. Although learning models on weakly labeled data might not be novel, improving deep CNNs on corrupted labels is clearly an under-studied problem and worthy of exploration, as deep CNNs are more prone to overfitting and memorizing corrupted labels (Zhang et al., 2017a). To address this issue, we focus on training very deep CNNs from scratch, such as resnet-101 (He et al., 2016) or inception-resnet (Szegedy et al., 2017) which has a few hundred layers and orders-of-magnitude more parameters than the number of training samples. These networks can achieve the state-of-the-art result but perform poorly when trained on corrupted labels.

Inspired by the recent success of Curriculum Learning (CL), this paper tackles this problem using CL (Bengio et al., 2009), a learning paradigm inspired by the cognitive process of human and animals, in which a model is learned gradually using samples ordered in a meaningful sequence. A curriculum specifies a scheme under which training samples will be gradually learned. CL has successfully improved the performance on a variety of problems. In our problem, our intuition is that a curriculum, similar to its role in education, may provide meaningful supervision to help a student overcome corrupted labels. A reasonable curriculum can help the student focus on the samples whose labels have a high chance of being correct.

However, for the deep CNNs, we need to address two limitations of the existing CL methodology. First, existing curriculums are usually predefined and remain fixed during training, ignoring the feedback from the student. The learning procedure of deep CNNs is quite complicated, and may not be accurately modeled by the predefined curriculum. Second, the alternating minimization, commonly used in CL and self-paced learning (Kumar et al., 2010) requires alternative variable updates, which is difficult for training very deep CNNs via mini-batch stochastic gradient descent.

To this end, we propose a method to learn the curriculum from data by a network called *MentorNet*. MentorNet learns a data-driven curriculum to supervise the base deep CNN, namely *StudentNet*. MentorNet can be learned to approximate an existing predefined curriculum or discover new data-driven curriculums from data. The learned data-driven

[1]Google Inc., Mountain View, United States [2]Stanford University, Stanford, United States. Correspondence to: Lu Jiang <lujiang@google.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

curriculum can be updated a few times taking into account of the StudentNet's feedback. Whenever MentorNet is learned or updated, we fix its parameter and use it together with StudentNet to minimize the learning objective, where MentorNet controls the timing and attention to learn each sample. At the test time, StudentNet makes predictions alone without MentorNet.

The proposed method improves existing curriculum learning in two aspects. First, our curriculum is learned from data rather than predefined by human experts. It takes into account of the feedback from StudentNet and can be dynamically adjusted during training. Intuitively, this resembles a "collaborative" learning paradigm, where the curriculum is determined by the teacher and student together. Second, in our algorithm, the learning objective is jointly minimized using MentorNet and StudentNet via mini-batch stochastic gradient descent. Therefore, the algorithm can be conveniently parallelized to train deep CNNs on big data. We show the convergence and empirically verify it on large-scale benchmarks.

We verify our method on four benchmarks. Results show that it can significantly improve the performance of deep CNNs trained on both controlled and real-world corrupted training data. Notably, to the best of our knowledge, it achieves the best-published result on WebVision (Li et al., 2017a), a large benchmark containing 2.2 million images of real-world noisy labels. To summarize, the contribution of this paper is threefold:

- We propose a novel method to learn data-driven curriculums for deep CNNs trained on corrupted labels.
- We discuss an algorithm to perform curriculum learning for deep networks via mini-batch stochastic gradient descent.
- We verify our method on 4 benchmarks and achieve the best-published result on the WebVision benchmark.

2. Preliminary on Curriculum Learning

We formulate our problem based on the model in (Kumar et al., 2010) and (Jiang et al., 2015). Consider a classification problem with the training set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$, where $\mathbf{x}_i$ denotes the i^{th} observed sample and $\mathbf{y}_i \in \{0,1\}^m$ is the noisy label vector over m classes. Let $g_s(\mathbf{x}_i, \mathbf{w})$ denote the discriminative function of a neural network called *StudentNet*, parameterized by $\mathbf{w} \in \mathbb{R}^d$. Further, let $\mathbf{L}(\mathbf{y}_i, g_s(\mathbf{x}_i, \mathbf{w}))$, a m-dimensional column vector, denote the loss over m classes. Introduce the latent weight variable, $\mathbf{v} \in \mathbb{R}^{n\times m}$, and optimize the objective:

$$\min_{\mathbf{w}\in\mathbb{R}^d, \mathbf{v}\in[0,1]^{n\times m}} \mathbb{F}(\mathbf{w},\mathbf{v}) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^T \mathbf{L}(\mathbf{y}_i, g_s(\mathbf{x}_i,\mathbf{w})) + G(\mathbf{v};\lambda) + \theta\|\mathbf{w}\|_2^2 \quad (1)$$

where $\|\cdot\|_2$ is the l_2 norm for weight decay, and data augmentation and dropout are subsumed inside g_s. $\mathbf{v}_i \in [0,1]^{m\times 1}$ is a vector to represent the latent weight variable for the i-th sample. The function G defines a *curriculum*, parameterized by λ. This paper focuses on the one-hot label. For notation convenience, denote the loss $\mathbf{L}(\mathbf{y}_i, g_s(\mathbf{x}_i,\mathbf{w})) = \ell_i$, $\mathbf{v}_i$ as a scalar v_i, and $\mathbf{y}_i$ as an integer $y_i \in [1, m]$.

In the existing literature, alternating minimization (Csiszar, 1984), or its related variants, is commonly employed to minimize the training objective, e.g. in (Kumar et al., 2010; Ma et al., 2017a; Jiang et al., 2014). This is an algorithmic paradigm where $\mathbf{w}$ and $\mathbf{v}$ are alternatively minimized, one at a time while the other is held fixed. When $\mathbf{v}$ is fixed, the weighted loss is typically minimized by stochastic gradient descent. When $\mathbf{w}$ is fixed, we compute $\mathbf{v}^k = \arg\min_{\mathbf{v}} \mathbb{F}(\mathbf{v}^{k-1}, \mathbf{w}^k)$ using the most recently updated $\mathbf{w}^k$ at epoch k. For example, Kumar *et al.* (2010) employed $G(\mathbf{v}) = -\lambda\|\mathbf{v}\|_1$. When $\mathbf{w}$ is fixed, the optimal $\mathbf{v}$ can be easily derived by:

$$v_i^* = \mathbb{1}(\ell_i \leq \lambda), \forall i \in [1, n], \quad (2)$$

where $\mathbb{1}$ is the indicator function. Eq. (2) intuitively explains the predefined curriculum in (Kumar et al., 2010), known as self-paced learning. First, when updating $\mathbf{v}$ with a fixed $\mathbf{w}$, a sample of smaller loss than the threshold λ is treated as an "easy" sample, and will be selected in training ($v_i^* = 1$). Otherwise, it will not be selected ($v_i^* = 0$). Second, when updating $\mathbf{w}$ with a fixed $\mathbf{v}$, the classifier is trained only on the selected "easy" samples. The hyperparameter λ controls the learning pace and corresponds to the "age" of the model. When λ is small, only samples of small loss will be considered. As λ grows, more samples of larger loss will be gradually added to train a more "mature" model.

As shown, the function G specifies a curriculum, *i.e.*, a sequence of samples with their corresponding weights to be used in training. When $\mathbf{w}$ is fixed, its optimal solution, *e.g.* Eq. (2), computes the time-varying weight that controls the timing and attention to learn every sample. Recent studies discovered multiple predefined curriculums and verified them in many real-world applications, *e.g.*, in (Fan et al., 2017; Ma et al., 2017a; Sangineto et al., 2016; Fan et al., 2017; Chang et al., 2017).

This paper studies learning curriculum from data. In the rest of this paper, Section 3 presents an approach to learn data-driven curriculum by *MentorNet*. Section 4 discusses an algorithm to optimize Eq. (1) using MentorNet and StudentNet together via mini-batch training.

3. Learning Curriculum from Data

Existing curriculums are either predetermined as an analytic expression of G or a function to compute sample weights. Such predefined curriculums cannot be adjusted accordingly, taking into account of the feedback from the student. This

section discusses a new way to learn data-driven curriculum by a neural network, called *MentorNet*. The MentorNet g_m is learned to compute time-varying weights for each training sample. Let Θ denote the parameters in g_m. Given a fixed $\mathbf{w}$, our goal is to learn an Θ^* to compute the weight:

$$g_m(\mathbf{z}_i;\Theta^*) = \arg\min_{v_i\in[0,1]} \mathbb{F}(\mathbf{w},\mathbf{v}), \forall i \in [1,n] \quad (3)$$

where $\mathbf{z}_i = \phi(\mathbf{x}_i, y_i, \mathbf{w})$ indicates the input feature to MentorNet about the i-th sample.

3.1. Learning Curriculum

MentorNet can be learned to 1) approximate existing curriculums or 2) discover new curriculums from data.

Learning to approximate predefined curriculums. Our first task is to learn a MentorNet to approximate a predefined curriculum. To do so, we minimize the objective in Eq. (1):

$$\arg\min_{\Theta} \sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}} g_m(\mathbf{z}_i;\Theta)\ell_i + G(g_m(\mathbf{z}_i;\Theta);\lambda) \quad (4)$$

Eq. (4) applies for both convex and non-convex G. This paper employs the following predefined curriculum. It is derived from (Jiang et al., 2015) and works well in our experiments. As will be discussed later, it is also related to robust non-convex penalties.

$$G(\mathbf{v};\lambda) = \sum_{i=1}^{n} \frac{1}{2}\lambda_2 v_i^2 - (\lambda_1+\lambda_2)v_i, \quad (5)$$

where $\lambda_1, \lambda_2 \geq 0$ are hyper-parameters. As G is convex, there exists a closed-form solution for the optimal value of Eq. (3). Given a fixed $\mathbf{w}$, define $\mathbb{F}_{\mathbf{w}}(\mathbf{v}) = \sum_{i=1}^{n} f(v_i)$:

$$f(v_i) = v_i\ell_i + \frac{1}{2}\lambda_2 v_i^2 - (\lambda_1+\lambda_2)v_i \quad (6)$$

The minima are obtained at $\nabla_{\mathbf{v}}\mathbb{F}_{\mathbf{w}}(\mathbf{v}) = 0$, and can be decoupled by setting $\partial f/\partial v_i = 0$. We then have:

$$g_m(\mathbf{z}_i;\Theta^*) = \begin{cases} \mathbb{1}(\ell_i \leq \lambda_1) & \lambda_2 = 0 \\ \min(\max(0, 1-\frac{\ell_i-\lambda_1}{\lambda_2}), 1) & \lambda_2 \neq 0 \end{cases}, \quad (7)$$

where Θ^* is the optimal MentorNet parameter obtained by SGD. The closed-form solution in Eq. (7) gives some intuitions about the curriculum. When $\lambda_2 = 0$, it is similar to self-paced learning (Kumar et al., 2010) *i.e.* only "easy" samples of $\ell_i < \lambda_1$ will be selected in training ($g_m(\mathbf{z}_i;\Theta^*) = 1$). When $\lambda_2 \neq 0$, samples of loss $\ell_i \geq \lambda_2 + \lambda_1$ will not be selected in training. These samples represent the "hard" samples of greater loss. Otherwise, samples will be weighted linearly w.r.t. $1-(\ell_i-\lambda_1)/\lambda_2$. As in (Kumar et al., 2010), the hyper-parameters λ_1 and λ_2 control the learning pace.

Learning data-driven curriculums. Our next task is to learn a curriculum solely derived from labeled data. To this end, Θ is learned on another dataset $\mathcal{D}' = \{(\phi(\mathbf{x}_i, y_i, \mathbf{w}), v_i^*)\}$, where $(\mathbf{x}_i, y_i)$ is sampled from $\mathcal{D}$ and $|\mathcal{D}'| \ll |\mathcal{D}|$. v_i^* is a given annotation and we assume it approximates the optimal weight, *i.e.*, $v_i^* \simeq \arg\min_{v_i\in[0,1]} \mathbb{F}(\mathbf{v},\mathbf{w})$. In this paper, we assign binary labels to v_i^*, where $v_i^* = 1$ iff y_i is a correct label. As v_i^* is binary, Θ is learned by minimizing the cross-entropy loss between v_i^* and $g(\mathbf{z}_i;\Theta)$. Intuitively, this process is similar to a mock test for the teacher (MentorNet) to learn to update her teaching strategy (curriculum). The student (StudentNet) provides features $\phi(\cdot,\cdot,\mathbf{w})$ for the mock test using the latest model $\mathbf{w}$. The teacher can learn an updated curriculum from the data to better supervise the latest student model. The learned curriculum is jointly determined by the teacher and student together.

The information on the correct label may not always be available on the target dataset $\mathcal{D}$. In this case, we learn the curriculum on a different small dataset where the correct labels are available. Intuitively, it resembles first learning a teaching strategy with the student on one topic and transfer the strategy on a similar topic. Empirically, Section 5.1 substantiates that the learned curriculum on a small subset of CIFAR-10 can be applied to the target CIFAR-100 dataset.

A burn-in period is introduced before learning Θ. In the first 20% training epoch of the StudentNet, MentorNet is initialized and fixed as $g_m(\mathbf{z}_i;\Theta^*) = r_i$, where $r_i \sim \text{Bernoulli}(p)$ is the Bernoulli random variable. This is equivalent to randomly dropping out p% training samples. We found that the burn-in process helps StudentNet stabilize the prediction and focus on learning simple and common patterns.

MentorNet architecture. We found that MentorNet can have a simple architecture. Appendix D shows that even MentorNet based on the two-layer perceptron can reasonably approximate the existing curriculum in the literature. Nevertheless, we use a MentorNet architecture shown in Fig. 1, which works reasonably well compared to classical network architectures. It takes the input of a mini-batch of samples, and outputs their corresponding sample weights. The feature $\mathbf{z}_i = \phi(\mathbf{x}_i, y_i, \mathbf{w})$ includes the loss, loss difference to the moving average, label and epoch percentage. ℓ_{pt} maintains an exponential moving average on the p-th percentile of the loss in each mini-batch. For a sample, its loss ℓ and loss difference $\ell - \ell_{pt}$ over the last few epochs can be encoded by a bidirectional LSTM network to capture the prediction variance (Chang et al., 2017). We verify the LSTM encoder in the experiments in Appendix D. For simplicity, we set the step size of the LSTM to 1 in Section 5.1 and only consider the loss and the loss difference of the current epoch.

The label and the training epoch percentage are encoded by two separate embedding layers. The epoch percentage is represented as an integer between 0 and 99. It is used to indicate the StudentNet's training progress, where 0 rep-

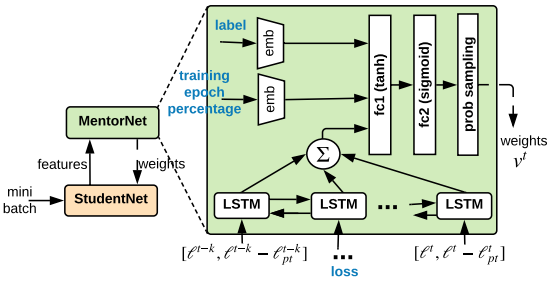


Figure 1. The MentorNet architecture used in experiments. *emb*, *fc* and *prob sampling* stand for the embedding, fully-connected and probabilistic sampling layer.

resents the first and 99 represents the last training epoch. The concatenated outputs from the LSTM and the embedding layers are fed into two fully-connected layers fc_1, fc_2, where fc_2 uses the sigmoid activation to ensure the output weights bounded between 0 and 1. The last layer in Fig. 1 is a probabilistic sampling layer, and is used to implement the sample dropout in the burn-in process on the already learned MentorNet.

3.2. Discussions

MentorNet is a general framework for both predefined and data-driven curriculum learning, where various curriculums can be learned by the same MentorNet structure with different parameters. This framework is conceptually general and practically flexible as we can switch curriculums by attaching different MentorNets without modifying the pipeline. Therefore, we also learn MentorNets for predefined curriculums. For predefined curriculums where G is unknown, we directly minimize the error between the MentorNet's outputs and desired weights. For example, the desired weight for focal loss (Lin et al., 2017b) is computed by:

$$v_i^* = [1 - \exp\{-\ell_i\}]^\gamma, \tag{8}$$

where γ is a hyperparameter for smoothing the distribution.

This paper tackles the problem of overcoming corrupted labels. It is interesting to analyze why the learned curriculum can improve the generalization performance. It turns out that StudentNet, when jointly learned with MentorNet, may optimize an underlying robust objective and the objective is also related to the robust M-estimator (Huber, 2011).

To show this, let $v^*(\lambda, x)$ represent the optimal weight function for a loss variable x, and we define:

$$v^*(\lambda, x) = \text{argmin}_{v\in[0,1]} \; vx + G(v, \lambda). \tag{9}$$

As g_m is an approximator to Eq. (9), its property can then be analyzed by the function $v^*(\lambda, x)$. Meng *et al.*(2015) investigated the insights of self-paced objective function, and proved that the optimization of SPL algorithm is intrinsically equivalent to minimizing a robust loss function. They showed that given a fixed λ and a decreasing $v^*(\lambda, x)$ with respect to x, the underlying objective of Eq. (1) can be obtained by:

$$F_\lambda(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\int_0^{\ell_i} v^*(\lambda, x)dx, \tag{10}$$

Based on it, the underlying learning objective of the curriculum in Eq. (5) can then be derived.

Remark 1. *When λ_1, λ_2 are fixed and $\lambda_2 \neq 0$, the underlying objective function of the curriculum in Eq. (5) is calculated from:*

$$F_\lambda(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases}\ell_i & \ell_i \le \lambda_1 \\ (\lambda_2 + 2\lambda_1)/2 & \ell_i \ge \lambda_2 + \lambda_1 \\ \theta\ell_i - \ell_i^2/(2\lambda_2) - \frac{(\theta-1)^2\lambda_2}{2} & \text{otherwise}\end{cases} \tag{11}$$

where $\theta = (\lambda_2 + \lambda_1)/\lambda_2$. When $\theta = 1$ it is equivalent to the minimax concave penalty (Zhang, 2010).

As shown in Eq. (11), the underlying objective has a form of $F_\lambda(\mathbf{w}) = \sum_i \rho(\ell_i)/n$, where ρ is the penalty function in M-estimator (Candes et al., 2008). Particularly, when $\theta = 1$, $\rho(\ell)$ is equivalent to the minimax concave plus penalty (Zhang, 2010), a popular non-convex robust loss. The result indicates the learned MentorNet that approximates our predefined curriculum in Eq. (5) leads to an underlying robust objective of the StudentNet.

For the data-driven curriculum, if the learned MentorNet satisfies certain conditions, we have:

Proposition 1. *Suppose $(\mathbf{x}, y)$ denotes a training sample and its corrupted label. For simplicity, let the MentorNet input $\phi(\mathbf{x}, y, \mathbf{w}) = \ell$ be the loss computed by the StudentNet model parameter $\mathbf{w}$. The MentorNet $g_m(\ell; \Theta) = v$, where v is the sample weight. If g_m decreases with respect to ℓ, then there exists an underlying robust objective F:*

$$F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\rho(\ell_i),$$

where $\rho(\ell_i) = \int_0^{\ell_i} g_m(x; \Theta)dx$. In the special cases, $\rho(\ell)$ degenerates to the robust M-estimator: Huber (Huber et al., 1964) and the log-sum penalty (Candes et al., 2008).

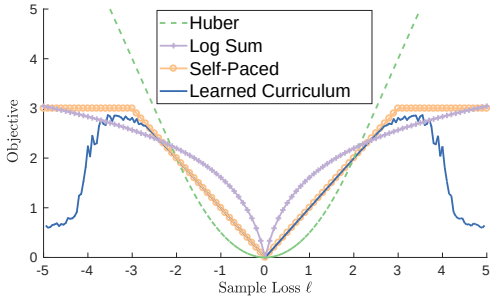


Figure 2. Visualization of sample loss and learning objective. The learned curriculum represents the best data-driven curriculum found in experiments.

The proposition indicates that there exist some learned MentorNets that are related to the robust M-estimator. On noisy

data, the effect of the robust objective is evident, *i.e.*, preventing StudentNet from being dominated by corrupted labels. Fig. 2 visualizes curves of the sample loss $\ell = y_i - g_s(\mathbf{x}_i, \mathbf{w})$ and the learning objective for the Huber loss (Huber et al., 1964), log-sum penalty (Candes et al., 2008), self-paced (Kumar et al., 2010), and our learned data-driven curriculum. We use the best learned curriculum Θ^* on CIFAR-10 in our experiments and plot $|g_m(\phi(\mathbf{x}, y, \mathbf{w}); \Theta^*) \times \ell|$ since the G in the objective function is unknown. As shown, all curves are robust to great loss to different extents. The corrupted labels in our problem are harmful. As the sample loss grows bigger beyond some value, MentorNet starts to sharply decrease the sample's weight. The subtlety of learned curriculum is difficult to be predefined by the analytic expression. Proposition 1 does not guarantee there is an underlying robust objective for every learned MentorNet. Instead, it shows MentorNet's capability of learning such robust objective.

4. The Algorithm

The alternating minimization algorithm (Csiszar, 1984) used in related work is intractable for deep CNNs, especially on big datasets, for two important reasons. First, in the subroutine of minimizing $\mathbf{w}$ when fixing $\mathbf{v}$, stochastic gradient descent often takes many steps before converging. This means that it can take a long time before moving past this single sub-step. However, such computation is often wasteful, particularly in the initial part of training, because, when $\mathbf{v}$ is far away from the optimal point, there is not much gain in finding the exact optimal $\mathbf{w}$ corresponding to this $\mathbf{v}$. Second, the subroutine of minimizing $\mathbf{v}$ when fixing $\mathbf{w}$ is often difficult, because the fixed vector $\mathbf{v}$ may not only consume a considerable amount of the memory but also hinder the parallel training on multiple machines. Therefore, optimizing the objective with deep CNNs requires some thought on the algorithmic level.

To minimize Eq. (1), we propose an algorithm called *SPADE* (Scholastic gradient PArtial DEscent). The algorithm optimizes the StudentNet model parameter $\mathbf{w}$ jointly with a given MentorNet. It provides a simple and elegant way to minimize $\mathbf{w}$ and $\mathbf{v}$ stochastically over mini-batches. As a general approach, it can also take an input of G. Let $\Xi_t = \{(\mathbf{x}_j, y_j)\}_{j=1}^b$ denotes a mini-batch of b samples, fetched uniformly at random and $\mathbf{v}_\Xi^t = [v_1^t, ..., v_b^t]$ represent the sample weights in Ξ_t. The MentorNet computes:

$$\mathbf{v}_\Xi^t = g_m(\phi(\Xi_t, \mathbf{w}^{t-1})) = \arg\min_{\mathbf{v}_\Xi} \mathbb{F}(\mathbf{w}^{t-1}, \mathbf{v}^{t-1}), \quad (12)$$

where ϕ is the feature extraction function defined in Eq. (3). Θ denotes the learned MentorNet discussed in Section 3.1.

As shown in Algorithm 1, for $\mathbf{w}$, a stochastic gradient is computed (via a mini-batch) and applied (Step 12), where α_t is the learning rate. For the latent weight variables $\mathbf{v}$, gradient descent is only applied to a small subset thereof parameters corresponding only to the mini-batch (Step 9 or 11). The partial gradient update on weight parameters is performed when G is used (Step 9). Otherwise, we directly apply the weights computed by the learned MentorNet (Step 11). In both cases, the weights are computed on-the-fly within a mini-batch and thus do not need to be fixed. As a result, the algorithm can be conveniently parallelized across multiple machines.

Algorithm 1 SPADE for minimizing Eq. (1)

Input : Dataset $\mathcal{D}$, a predefined G or a learned $g_m(\cdot; \Theta)$
Output : The model parameter $\mathbf{w}$ of StudentNet.

1 Initialize $\mathbf{w}^0, \mathbf{v}^0, t = 0$
2 **while** *Not Converged* **do**
3 Fetch a mini-batch Ξ_t uniformly at random
4 For every $(\mathbf{x}_i, y_i)$ in Ξ_t compute $\phi(\mathbf{x}_i, y_i, \mathbf{w}^t)$
5 **if** *update curriculum* **then**
6 $\Theta \leftarrow \Theta^*$, where Θ^* is learned in Sec. 3.1
7 **end**
8 **if** *G is used* **then**
9 $\mathbf{v}_\Xi^t \leftarrow \mathbf{v}_\Xi^{t-1} - \alpha_t \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t-1}, \mathbf{v}^{t-1})|_{\Xi_t}$
10 **end**
11 **else** $\mathbf{v}_\Xi^t \leftarrow g_m(\phi(\Xi_t, \mathbf{w}^{t-1}); \Theta)$;
12 $\mathbf{w}^t \leftarrow \mathbf{w}^{t-1} - \alpha_t \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^{t-1}, \mathbf{v}^t)|_{\Xi_t}$
13 $t \leftarrow t + 1$
14 **end**
15 **return** $\mathbf{w}^t$

The curriculum can change during training. MentorNet is updated a few times in Algorithm 1. In Step 6, the MentorNet parameter Θ is updated to adapt to the most recent model parameters of StudentNet. In experiments, we update Θ twice after the learning rate is changed. Each time, a data-driven curriculum is learned from the data generated by the most recent $\mathbf{w}$ using the method discussed in Section 3.1. The update is consistent with existing curriculum learning methodology (Bengio et al., 2009; Kumar et al., 2010) and the difference here is that for each update, the curriculum is learned rather than specified by human experts.

Under standard assumptions, Theorem 1 shows that the algorithm stabilizes and converges to a stationary point (convergence to global/local minima cannot be guaranteed unless in specially structured non-convex objectives (Chen et al., 2018; Zhou et al., 2017b;a)). The proof is in Appendix B. The theorem is a characterization of stability of the model parameters $\mathbf{w}$. For the weight parameters $\mathbf{v}$, as it is restricted in a compact set, convergence to a stationary point is not always guaranteed. As the model parameters are more important, we only provide a detailed characterization of the model parameter.

Theorem 1. *Let the objective $\mathbb{F}(\mathbf{w}, \mathbf{v})$ defined in Eq. (1) be differentiable, $L(\cdot)$ be Lipschitz continuous in $\mathbf{w}$ and $\nabla_{\mathbf{v}} G(\cdot)$ be Lipschitz continuous in $\mathbf{v}$. Let $\mathbf{w}^t, \mathbf{v}^t$ be iterates from Algorithm 1 and $\sum_{t=0}^{\infty} \alpha_t = \infty, \sum_{t=0}^{\infty} \alpha_t^2 < \infty$. Then, $\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)\|_2^2] = 0$.*

For the manually designed curriculums, it may be unclear

where or even whether such predefined curriculum would converge via mini-batch training. Theorem 1 shows that the learned curriculum can converge and produce a stable StudentNet model. The algorithm can be used to replace the alternating minimization method in related work.

5. Experiments

This section empirically verifies the proposed method on four benchmarks of controlled corrupted labels in Section 5.1 and real-world noisy labels in Section 5.2. The code can be found at https://github.com/google/mentornet.

5.1. Experiments on controlled corrupted labels

This section validates MentorNet on the controlled corrupted label. We follow a common setting in (Zhang et al., 2017a) to train deep CNNs, where the label of each image is independently changed to a uniform random class with probability p, where p is noise fraction and is set to 0.2, 0.4 and 0.8. The labels of validation data remain clean for evaluation.

Dataset and StudentNet: We use the same benchmarks in (Zhang et al., 2017a): CIFAR-10, CIFAR-100 and ImageNet. CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009) consist of 32×32 color images arranged in 10 and 100 classes. Both datasets contain 50,000 training and 10,000 validation images. ImageNet ILSVRC2012 (Deng et al., 2009) contain about 1.2 million training and 50k validation images, split into 1,000 classes. Each image is resized to 299x299 with 3 color channels.

We employ 3 recent deep CNNs as our StudentNets: inception (Szegedy et al., 2016), resnet-101 (He et al., 2016) with wide filters (Zagoruyko & Komodakis, 2016) and inception-resnet v2 (Szegedy et al., 2017). Table 1 shows their #model parameters, training, and validation accuracy when we train them on the clean training data (noise$= 0$). As shown, they achieve reasonable accuracy on each task.

Table 1. StudentNet and their accuracies on the clean training data.

Dataset	Model	#para	train acc	val acc
CIFAR10	inception	1.7M	0.83	0.81
	resnet101	84M	1.00	0.96
CIFAR100	inception	1.7M	0.64	0.49
	resnet101	84M	1.00	0.79
ImageNet	inception_resnet	59M	0.88	0.77

Baselines: MentorNet is compared against the following baselines: **FullMode** is the standard StudentNet trained using l_2 weight decay, dropout (Srivastava et al., 2014) and data augmentation (Krizhevsky et al., 2012). The hyperparameters are set to the best ones found on the clean training data. Unless specified otherwise, for a fair comparison, the StudentNet with the same hyperparameters is used in all baseline and our model. **Forgetting** was introduced in (Arpit et al., 2017), in which the dropout parameter is searched in the range of (0.2-0.9). **Self-paced** (Kumar et al., 2010) and **Focal Loss** (Lin et al., 2017b) represent well-known predefined curriculums in the literature. We implemented **Reed** (2014) and **Goldberger** (Goldberger & Ben-Reuven, 2017) as the recent weakly-supervised learning methods. The above baseline methods are a mixture of the curriculum learning and the recent methods dealing with corrupted labels.

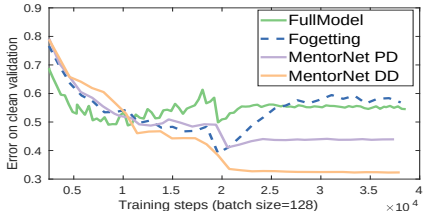
Our Model: **MentorNet PD** is the network learned using our *predefined* curriculum in Eq. (5) using no additional clean labels. **MentorNet DD** is the learned *data-driven* curriculum. It is trained on 5,000 images of true labels, randomly sampled from the CIFAR-10 training set. The same data are used to learn MentorNet DD on CIFAR-100. Note CIFAR-10 and CIFAR-100 are two different datasets that have not only different classes but also the different number of classes. Therefore, it is fair to compare MentorNet DD with other methods using no true labels on CIFAR-100. Algorithm 1 is used to optimize the StudentNet. The decay factor in computing the loss moving average is set to 0.95. The loss percentile in the moving average is set by the cross-validation. As mentioned, a burn-in process is used in the first 20% training epoch for both MentorNet DD and MentorNet PD. More details are discussed in Appendix E.

We first show the comparison to the baseline method on CIFAR-10 and CIFAR-100 in Table 2. On both datasets, each method is verified with two StudentNets (resnet-101 and inception) under the noise fraction of 0.2, 0.4, and 0.8. As we see on both datasets, MentorNet improves FullModel across different noise fractions, and the learned data-driven curriculum (MentorNet DD) achieves the best results. The improvement is more significant for the deeper CNN model resnet-101. For example, on the CIFAR-10 of 40% noise, MentorNet DD (with resnet-101) yields an absolute 20% gain over FullModel. After inspecting the result, we found that it may be because Mentor DD learns a more appropriate curriculum to give high weights to samples of correct labels. As a result, it helps the StudentNet focus on samples of correct labels. The results indicate that the learned MentorNet can improve the generalization performance of recent deep CNNs, and outperform the predefined curriculums (Self-paced and Focal Loss).

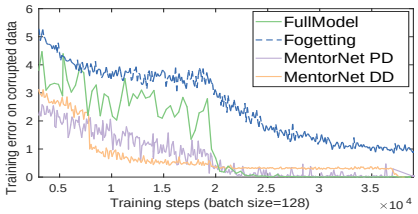
Fig. 3 plots the training and test error on the clean validation data, under a representative setting: resnet-101 on CIFAR-100 of 40% noise, where the x-axis denotes the training iteration. The y-axis is the validation error on the clean validation in Fig. 3(a) and the mini-batch training error on corrupted labels in Fig. 3(b). For MentorNet, the training error is computed by $\sum_i v_i \ell_i$. The figure shows two insights. First, the training error of MentorNet approaches zero. This empirically verifies the convergence of the model. Second, MentorNet can overcome the overfitting to the corrupted label. While the training error is decreasing, the test error

Table 2. Comparison of validation accuracy on CIFAR-10 and CIFAR-100 under different noise fractions.

	Resnet-101 StudentNet						Inception StudentNet					
	CIFAR-100			CIFAR-10			CIFAR-100			CIFAR-10		
Method	0.2	0.4	0.8	0.2	0.4	0.8	0.2	0.4	0.8	0.2	0.4	0.8
FullModel	0.60	0.45	0.08	0.82	0.69	0.18	0.43	0.38	0.15	0.76	0.73	0.42
Forgetting	0.61	0.44	0.16	0.78	0.63	0.35	0.42	0.37	0.17	0.76	0.71	0.44
Self-paced	0.70	0.55	0.13	0.89	0.85	0.28	0.44	0.38	0.14	**0.80**	0.74	0.33
Focal Loss	0.59	0.44	0.09	0.79	0.65	0.28	0.43	0.38	0.15	0.77	0.74	0.40
Reed Soft	0.62	0.46	0.08	0.81	0.63	0.18	0.42	0.39	0.12	0.78	0.73	0.39
MentorNet PD	0.72	0.56	0.14	0.91	0.77	0.33	0.44	0.39	0.16	0.79	0.74	0.44
MentorNet DD	**0.73**	**0.68**	**0.35**	**0.92**	**0.89**	**0.49**	**0.46**	**0.41**	**0.20**	0.79	**0.76**	**0.46**



(a) Test error on clean validation



(b) Training error on corrupted labels

Figure 3. Error of resnet trained on CIFAR-100 of 40% noise.

does not increase in Fig. 3(a). It suggests that the learned curriculum is beneficial for StudentNet. The sharp change around the 20k iteration in Fig. 3 is due to the learning rate change. Besides, our result is consistent with (Zhang et al., 2017a) that deep CNNS is able to get 0 training error on the corrupted training data. Forgetting (the dashed curve) is the only one that does not converge within 30k steps. As indicated in (Arpit et al., 2017), it is because forgetting reduces the speed at which DNNs memorize. As suggested in (Zhang et al., 2017b), a *not* converged model might yield a better result, *e.g.*, stop the model at 20K in Fig. 3. However, as it is hard to predetermine the time for early stopping, our focus is comparing the converged model.

Fig. 4 illustrates the best learned data-driven curriculum in our experiments, where the z-axis denotes the weights computed by g_m; the y and x axes denote the sample loss and the loss difference to the moving average, where λ is the loss moving average. Two observations can be found in Fig. 4. First, the learned curriculum changes during the training of the StudentNet. Fig. 4 (a) and (b) are MentorNet learned at different epochs. As shown, (a) assigns greater weights to samples of big loss more aggressively. Second, the learned curriculums in Fig. 4 generally satisfy the condition in Proposition 1, *i.e.*, the weight generally decreases with the loss. It suggests that joint learning of StudentNet and MentorNet optimizes an underlying robust objective.

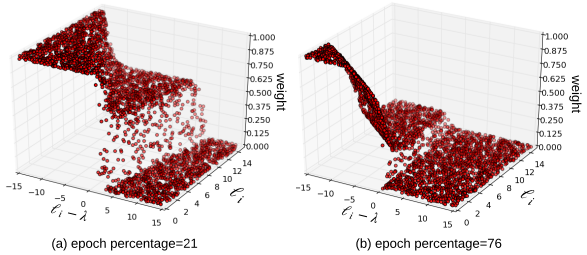


Figure 4. The data-driven curriculums learned by MentorNet with the resnet-101 at epoch 21 in (a) and 76 in (b).

Table 3 compares to recent published results under the setting: CIFAR of 40% noise fraction. We cite the number in (Azadi et al., 2016), and implement other methods using the same resnet-101 StudentNet. The results show that our result is comparable and even better than the state-of-the-art.

Table 3. Validation accuracy comparison to representative published results on CIFAR of 40% noise.

ID	Method	CIFAR-10	CIFAR-100
1	Reed Hard (Resnet)	0.62	0.47
2	Reed Soft (Resnet)	0.61	0.46
3	Goldberger (Resnet)	0.70	0.46
4	Azadi (AlexNet) (2016)	0.75	-
5	**MentorNet (Resnet)**	**0.89**	**0.68**

To verify MentorNet for large-scale training, we apply our method on the ImageNet ILSVRC12 (Deng et al., 2009) benchmark to improve the inception-resnet v2 (Szegedy et al., 2017) model. We train the model on the ImageNet of 40% noise. Inspired by (Zhang et al., 2017a), we start with an inception-resnet (NoReg) with no regularization (NoReg) and add weight decay, dropout, and data augmentation to the model. Table 4 shows the comparison. As shown, MentorNet improves the performance of both the inception-resnet without regularization (NoReg) and with full regularization (FullModel). It also outperforms the forgetting baseline (dropout keep probability = 0.2). The results suggest that MentorNet can improve deep CNNs on the large-scale training on corrupted labels.

5.2. Experiments on real-world noisy labels

To verify MentorNet on real-world noisy labels, we conduct experiments on the large WebVision benchmark (Li et al.,

Table 4. Accuracy on the clean ImageNet ImageNet validation set. Models are trained on the ImageNet training data of 40% noise.

Method	P@1	P@5
NoReg	0.538	0.770
NoReg+WeDecay	0.560	0.809
NoReg+Dropout	0.575	0.807
NoReg+DataAug	0.522	0.772
NoReg+MentorNet	0.590	0.814
FullModel	0.612	0.844
Forgetting(FullModel)	0.628	0.845
MentorNet(FullModel)	**0.651**	**0.859**

2017a). It contains 2.4 million images of real-world noisy labels, crawled from the web using the 1,000 concepts in ImageNet ILSVRC12. We download the resized images from the official website[1]. The inception-resenet v2 (Szegedy et al., 2017) is used as our StudentNet, trained using a distributed asynchronized momentum optimizer on 50 GPUs. Since the dataset is very big, for quick experiments, we compare baseline methods using the Google image subset on the first 50 classes. We use Mini to denote this subset and Entire for the entire WebVision. All the models are evaluated on the clean ILSVRC12 and WebVision validation set.

Table 5 lists the comparison result. As we see, the proposed MentorNet significantly improves baseline methods on real-world noisy labels. The method marked by the start indicates it uses a pre-trained ImageNet model to obtain additional 30k labels for 118 classes. Following the same protocol, MentorNet* is trained using the additional labels. The results show that our method outperforms the baseline methods on real-world noisy labels. To the best of our knowledge, it achieves the best-published result on the WebVision (Li et al., 2017a) benchmark.

Table 5. Validation accuracy on the ImageNet ILSVRC12 and WebVision validation set. The number outside (inside) the parentheses denotes top-1 (top-5) classification accuracy (%). * marks the method trained using additional verification labels.

Dataset	Method	ILSVRC12	WebVision
Entire	Li *et al.* (2017a)	0.476 (0.704)	0.570 (0.779)
Entire	Forgetting	0.590 (0.808)	0.666 (0.856)
Entire	Lee *et al.* (2017)*	0.602 (0.811)	0.685 (0.865)
Entire	MentorNet	0.625 (0.830)	0.708 (0.880)
Entire	**MentorNet***	**0.642** (**0.848**)	**0.726** (**0.889**)
Mini	FullModel	0.585 (0.818)	-
Mini	Forgetting	0.562 (0.816)	-
Mini	Reed Soft	0.565 (0.827)	-
Mini	Self-paced	0.576 (0.822)	-
Mini	**MentorNet**	**0.638** (**0.858**)	-

6. Related Work

Curriculum learning (CL), proposed by Bengio *et al.* (2009), is a learning paradigm in which a model is learned by gradually including from easy to complex samples in training so as to increase the learning entropy (Bengio et al., 2009). From the human behavioral perspective, Khan *et al.* (2011) have shown that CL is consistent with the principle of human teaching. CL has been empirically verified in a variety of problems, such as computer vision (Supancic & Ramanan, 2013; Chen & Gupta, 2015), natural language processing (Turian et al., 2010), multitask learning (Graves et al., 2017). A common CL approach is to predefine a curriculum. For example, Kumar *et al.* (2010) proposed a curriculum called self-paced learning which favors training samples of smaller loss. After that, many predefined curriculums were proposed, *e.g.*, in (Supancic & Ramanan, 2013; Jiang et al., 2014; 2015; Sangineto et al., 2016; Chang et al., 2017; Ma et al., 2017a;b). For example, Jiang *et al.* (2014) introduced a curriculum of using easy and diverse samples. Fan *et al.* (2017) proposed to use predefined sample weighting schemes as an implicit way to define a curriculum. Previous work has shown that predefined curriculums are useful in overcoming noisy labels (Chen & Gupta, 2015; Liang et al., 2016; Lin et al., 2017a). In parallel to CL, the sample weighting schemes were also studied in (Lin et al., 2017a; Wang et al., 2017; Fan et al., 2018; Dehghani et al., 2018). Compared to the existing work, our paper presents a new way of learning data-driven curriculums for deep networks trained on corrupted labels.

Our work is related to the weakly-supervised learning methods. Among recent contributions, Reed *et al.* (2014) developed a robust loss to model "prediction consistency". Menon *et al.* (2015) used class-probability estimation to study the corruption process. Sukhbaatar *et al.* (2014) proposed a noise transformation to estimate the noise distribution. The transformation matrix needs to be periodically updated and is non-trivial to learn. To address the issue, Goldberger *et al.* (2017) proposed to add an additional softmax layer end-to-end with the base model. Azadi *et al.* (2016) tackled this problem by a regularizer called AIR. This method was shown to be effective but it relied on additional clean labels to train the representation. More recently, methods utilized additional labels for label cleaning (Veit et al., 2017), knowledge distillation (Li et al., 2017b) or semi-supervised learning (Vahdat, 2017; Dehghani et al., 2017). Different from previous work, we focus on learning curriculum to train very deep CNNs on corrupted labels from scratch. In addition, clean labels are not always needed for our method. In Section 5.1, the MentorNet is learned on a small subset of CIFAR-10 and applied to CIFAR-100

7. Conclusions

In this paper, we presented a novel method for training deep CNNs on corrupted labels. Our work was built on curriculum learning and advanced the methodology by proposing to learn data-driven curriculum via a neural network called MentorNet. We proposed an algorithm for jointly optimizing deep CNNs with MentorNet on large-scale data. We conducted comprehensive experiments on datasets of controlled and real-world noise. Our empirical results showed

[1]https://www.vision.ee.ethz.ch/webvision/download.html

that generalization performance of deep CNNs trained on corrupted labels can be effectively improved by the learned data-driven curriculum.

Acknowledgements

The authors would like to thank anonymous reviewers for helpful comments and Deyu Meng, Sergey Ioffe, and Chong Wang for meaningful discussions and kind support.

References

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

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

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *ICML*, 2009.

Candes, E. J., Wakin, M. B., and Boyd, S. P. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

Chang, H.-S., Learned-Miller, E., and McCallum, A. Active bias: Training a more accurate neural network by emphasizing high variance samples. *NIPS*, 2017.

Chen, X. and Gupta, A. Webly supervised learning of convolutional networks. In *ICCV*, 2015.

Chen, Y., Chi, Y., Fan, J., and Ma, C. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. *arXiv preprint arXiv:1803.07726*, 2018.

Csiszar, I. Information geometry and alternating minimization procedures. *Statistics and decisions*, 1:205–237, 1984.

Dehghani, M., Severyn, A., Rothe, S., and Kamps, J. Avoiding your teacher's mistakes: Training neural networks with controlled weak supervision. *arXiv preprint arXiv:1711.00313*, 2017.

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

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Fan, Y., He, R., Liang, J., and Hu, B.-G. Self-paced learning: An implicit regularization perspective. In *AAAI*, 2017.

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

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

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. Automated curriculum learning for neural networks. In *ICML*, 2017.

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

Huber, P. J. Robust statistics. In *International Encyclopedia of Statistical Science*, pp. 1248–1251. Springer, 2011.

Huber, P. J. et al. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

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

Jiang, L., Meng, D., Zhao, Q., Shan, S., and Hauptmann, A. G. Self-paced curriculum learning. In *AAAI*, 2015.

Khan, F., Mutlu, B., and Zhu, X. How do humans teach: On curriculum learning and teaching dimension. In *NIPS*, 2011.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

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

Lee, K.-H., He, X., Zhang, L., and Yang, L. Cleannet: Transfer learning for scalable image classifier training with label noise. *arXiv preprint arXiv:1711.07131*, 2017.

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

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

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

Lin, L., Wang, K., Meng, D., Zuo, W., and Zhang, L. Active self-paced learning for cost-effective and progressive face identification. *TPAMI*, 2017a.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. *ICCV*, 2017b.

Ma, F., Meng, D., Xie, Q., Li, Z., and Dong, X. Self-paced co-training. In *ICML*, 2017a.

Ma, Z., Liu, S., and Meng, D. On convergence property of implicit self-paced objective. *arXiv preprint arXiv:1703.09923*, 2017b.

Meng, D., Zhao, Q., and Jiang, L. What objective does self-paced learning indeed optimize? *arXiv preprint arXiv:1511.06049*, 2015.

Menon, A., Van Rooyen, B., Ong, C. S., and Williamson, B. Learning from corrupted binary labels via class-probability estimation. In *ICML*, 2015.

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

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Sangineto, E., Nabi, M., Culibrk, D., and Sebe, N. Self paced deep learning for weakly supervised object detection. *arXiv preprint arXiv:1605.07651*, 2016.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., and Fergus, R. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

Supancic, J. S. and Ramanan, D. Self-paced learning for long-term tracking. In *CVPR*, 2013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.

Turian, J., Ratinov, L., and Bengio, Y. Word representations: a simple and general method for semi-supervised learning. In *ACL*, 2010.

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

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., and Belongie, S. 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.

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*, 2017a.

Zhang, C.-H. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, pp. 894–942, 2010.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017b.

Zhou, Z., Mertikopoulos, P., Bambos, N., Boyd, S., and Glynn, P. Mirror descent in non-convex stochastic programming. *arXiv preprint arXiv:1706.05681*, 2017a.

Zhou, Z., Mertikopoulos, P., Bambos, N., Boyd, S., and Glynn, P. W. Stochastic mirror descent in variationally coherent optimization problems. In *NIPS*, 2017b.

Supplementary Materials: MentorNet Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels

Lu Jiang Zhengyuan Zhou Thomas Leung Li-Jia Li Li Fei-Fei

A. Derivation of Remark 1

Our objective function is:

$$\min_{\mathbf{w}\in\mathbb{R}^d,\mathbf{v}\in[0,1]^n} \mathbb{F}(\mathbf{w},\mathbf{v}) = \frac{1}{n}\sum_{i=1}^{n} v_i \mathbf{L}(y_i, g_s(\mathbf{x}_i,\mathbf{w})) + G(\mathbf{v};\lambda) + \theta\|\mathbf{w}\|_2^2 \tag{1}$$

Let $\ell_i = \mathbf{L}(y_i, g_s(\mathbf{x}_i, \mathbf{w}))$ denote the loss of the i-th sample ($\ell_i \geq 0$). The predefined curriculum is defined as:

$$G(\mathbf{v};\lambda) = \sum_{i=1}^{n} \frac{1}{2}\lambda_2 v_i^2 - (\lambda_2 + \lambda_1)v_i. \tag{2}$$

Denote $\mathbb{F}_{\mathbf{w}}$ as the objective function when the $\mathbf{w}$ is fixed. We have

$$\begin{aligned}
\mathbb{F}_{\mathbf{w}}(\mathbf{v}) &= \frac{1}{n}\sum_{i=1}^{n} v_i\ell_i + G(\mathbf{v};\lambda) + \theta\|\mathbf{w}\|_2^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} v_i\ell_i + \frac{1}{2}\lambda_2 v_i^2 - (\lambda_2+\lambda_1)v_i + \theta\|\mathbf{w}\|_2^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} f(v_i) + \theta\|\mathbf{w}\|_2^2
\end{aligned} \tag{3}$$

where $f(v_i) = v_i\ell_i + \frac{1}{2}\lambda_2 v_i^2 - (\lambda_2+\lambda_1)v_i$.

As $f(v_i)$ is convex with respect to v_i ($\lambda_2 \geq 0$). Its minimum is obtained at

$$\begin{aligned}
&\arg\min_{\mathbf{v}\in[0,1]^n} \nabla_{\mathbf{v}}\mathbb{F}_{\mathbf{w}}(\mathbf{v}) = 0 \\
&\Rightarrow \frac{\partial f(v_i)}{\partial v_i} = \ell_i + \lambda_2 v_i - \lambda_2 - \lambda_1 = 0,\ (\forall i \in [1,n], v_i \in [0,1])
\end{aligned} \tag{4}$$

Since $\lambda_1, \lambda_2 \geq 0$ and v_i is bounded in $[0,1]$. When $\lambda_2 \neq 0$, the optimal v_i^* is calculated from:

$$v_i^* = \begin{cases} 1 & (\ell_i \leq \lambda_1) \wedge (\lambda_2 \neq 0) \\ 1 - \frac{\ell_i - \lambda_1}{\lambda_2} & (\lambda_1 < \ell_i < \lambda_2 + \lambda_1) \wedge (\lambda_2 \neq 0) \\ 0 & (\ell_i \geq \lambda_2 + \lambda_1) \wedge (\lambda_2 \neq 0) \end{cases}, \tag{5}$$

When $\lambda_2 = 0$, the optimal weight writes as:

$$v_i^* = \begin{cases} 1 & (\ell_i < \lambda_1) \wedge (\lambda_2 = 0) \\ 0 & (\ell_i \geq \lambda_1) \wedge (\lambda_2 = 0) \end{cases}. \tag{6}$$

Combined Eq. (5) and Eq. (6), we have

$$v_i^* = \begin{cases} \mathbb{1}(\ell_i \leq \lambda_1) & \lambda_2 = 0 \\ \min(\max(0, 1 - \frac{\ell_i - \lambda_1}{\lambda_2}), 1) & \lambda_2 \neq 0 \end{cases} \tag{7}$$

According to the definition of Θ, we have $g_m(\phi(\mathbf{x}_i, y_i, \mathbf{w}); \Theta^*) = v_i^*$. Incorporating Eq. (7), we have

$$g_m(\phi(\mathbf{x}_i, y_i, \mathbf{w}); \Theta^*) = \begin{cases} \mathbb{1}(\ell_i \leq \lambda_1) & \lambda_2 = 0 \\ \min(\max(0, 1 - \frac{\ell_i - \lambda_1}{\lambda_2}), 1) & \text{otherwise} \end{cases} \tag{8}$$

Now we derive its underlying objective. First, we define a function $v^*(\lambda, x)$ and incorporate the above optimal solution:

$$v^*(\lambda, x) = \arg\min_{v \in [0,1]} \; vx + G(v, \lambda) = \begin{cases} \mathbb{1}(x \leq \lambda_1) & \lambda_2 = 0 \\ \min(\max(0, 1 - \frac{x - \lambda_1}{\lambda_2}), 1) & \text{otherwise} \end{cases} \tag{9}$$

Let $\epsilon > 0$ denote a small positive constant. Using the condition $\lambda_2 \geq 0$, we incorporate Eq. (7):

$$v^*(\lambda, x + \epsilon) - v^*(\lambda, x) \leq 0 \tag{10}$$

That indicates $v^*(\lambda, x)$ decreases with respect to x. According to (Meng et al., 2015), given the fixed hyperparameter λ (*i.e.* λ_1, λ_2), its underlying objective function has the form of:

$$F_\lambda(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \int_0^{\ell_i} v^*(\lambda; x) dx, \tag{11}$$

After incorporating Eq. (5) and Eq. (6) into Eq. (11), we have when $\lambda_2 = 0$

$$F_\lambda(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \min(\ell_i, \lambda_1) \tag{12}$$

When $\lambda_2 \neq 0$, we have:

$$\begin{aligned} F_\lambda(\mathbf{w}) &= \frac{1}{n} \sum_{i=1}^{n} \begin{cases} \ell_i & \ell_i \leq \lambda_1 \\ \ell_i + \frac{\lambda_1}{\lambda_2} \ell_i - \frac{1}{2\lambda_2} \ell_i^2 - \frac{\lambda_1^2}{2\lambda_2} & \lambda_1 < \ell_i < \lambda_2 + \lambda_1 \\ \frac{\lambda_2 + 2\lambda_1}{2} & \ell_i \geq \lambda_2 + \lambda_1 \end{cases} \\ &= \frac{1}{n} \sum_{i=1}^{n} \begin{cases} \ell_i & \ell_i \leq \lambda_1 \\ \theta \ell_i - \ell_i^2 / (2\lambda_2) - \frac{(\theta - 1)^2 \lambda_2}{2} & \lambda_1 < \ell_i < \lambda_2 + \lambda_1 \\ (\lambda_2 + 2\lambda_1)/2 & \ell_i \geq \lambda_2 + \lambda_1 \end{cases} \end{aligned}, \tag{13}$$

where $\theta = (\lambda_2 + \lambda_1)/\lambda_2$ and $\lambda_1, \lambda_2 \geq 0$. We $\lambda_2 \neq 0$, we have the Eq.(10) in the Remark 1 in the paper.

$$F_\lambda(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \begin{cases} \ell_i & \ell_i \leq \lambda_1 \\ (\lambda_2 + 2\lambda_1)/2 & \ell_i \geq \lambda_2 + \lambda_1 \\ \theta \ell_i - \ell_i^2 / (2\lambda_2) - \frac{(\theta - 1)^2 \lambda_2}{2} & \text{otherwise} \end{cases}, \tag{14}$$

When $\theta = 1$ we have $\lambda_1 = 0$ and the above equation becomes:

$$F_\lambda(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \begin{cases} \ell_i - \ell_i^2 / (2\lambda_2) & \ell_i < \lambda_2 \\ \lambda_2 / 2 & \ell_i \geq \lambda_2 \end{cases}. \tag{15}$$

The above equation is equivalent to the minimax concave penalty (MCP) (Gong et al., 2013; Zhang, 2010). Below is its formula presented in (Gong et al., 2013) (with the regularization hyperparameter setting to 1):

$$\int_0^{\ell} [1 - \frac{x}{t}] dx = \begin{cases} \ell - \ell^2 / (2t) & \ell < t \\ t/2 & \ell \geq t \end{cases}, \tag{16}$$

B. Proof of Theorem 1

Theorem 1 *Let the objective $\mathbb{F}(\mathbf{w}, \mathbf{v})$ defined in Eq. (1) be differentiable, $L(\cdot)$ be Lipschitz continuous in $\mathbf{w}$ and $\nabla_{\mathbf{v}} G(\cdot)$ be Lipschitz continuous in $\mathbf{v}$. Let $\mathbf{w}^t, \mathbf{v}^t$ be iterates from Algorithm 1 and $\sum_{t=0}^{\infty} \alpha_t = \infty, \sum_{t=0}^{\infty} \alpha_t^2 < \infty$. Then, $\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)\|_2^2] = 0$.*

Proof 1 *First, by the definition of the objective function $\mathbb{F}(\mathbf{w}, \mathbf{v})$, it can be easily checked that $L(\cdot)$ being Lipschitz continuous in $\mathbf{w}$ implies that $\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}, \mathbf{v})$ is a Lipschitz function in $\mathbf{w}$ for every $\mathbf{v}$. Similarly, $\nabla_{\mathbf{v}} G(\cdot)$ being Lipschitz continuous in $\mathbf{v}$ implies that $\nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}, \mathbf{v})$ is a Lipschitz function in $\mathbf{v}$ for all $\mathbf{w}$. Further, without loss of generality, we assume all the Lipschitz constants are (upper bounded by) L.*

Throughout the proof, as in the main text, n is the size of the training dataset. Define the n-dimensional vector e^k as $e_i^t = 1$ if $(x_i, y_i) \in \Xi_t$ and 0 otherwise and denote by $\otimes$ the point-wise product operation between two vectors: $[a_1, a_2] \otimes [b_1, b_2] = [a_1 b_1, a_2 b_2]$. When G is used, the update in each iteration is two consecutive gradient steps as follows:

$$\begin{aligned} \mathbf{w}^{t+1} &= \mathbf{w}^t - \alpha_t \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)|_{\Xi_t}, \\ \mathbf{v}^{t+1} &= \mathbf{v}^t - \alpha_t \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t)|_{\Xi_t}. \end{aligned}$$

Since the mini-batch Ξ_t is draw uniformly at random, we can rewrite the update as:

$$\begin{aligned} \mathbf{w}^{t+1} &= \mathbf{w}^t - \alpha_t [\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^k, \mathbf{v}^t) + \xi^t], \\ \mathbf{v}^{t+1} &= \mathbf{v}^t - \alpha_t [e^t \otimes \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t)], \end{aligned}$$

*where $\xi^t = \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)|_{\Xi_t} - \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^k, \mathbf{v}^t)$. Note that both ξ^t and e^t are **iid** random variables with finite variance, since Ξ_t are drawn **iid** with a finite number (b) of samples. Further, $\mathbb{E}[\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)|_{\Xi_t} - \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^k, \mathbf{v}^t)] = 0$, since samples are drawn uniformly at random.*

By Lipschitz continuity of $\nabla_{\mathbf{w}} F(\mathbf{w}, \mathbf{v})$ (and L being the Lipschitz constant), we obtain the following:

$$\begin{aligned} &\mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) - \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t) \le \langle \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle + \frac{L}{2} \|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2 \\ &= \langle \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t), -\alpha_t [\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t) + \xi^t] \rangle + \frac{L}{2} \|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2 \\ &= -\alpha_t \{\|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)\|_2^2 + \langle \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t), \xi^t \rangle\} + \frac{L}{2} \|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2 \\ &= -\alpha_t \{\|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)\|_2^2 + \langle \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t), \xi^t \rangle\} + \frac{L\alpha_t^2}{2} \|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t) + \xi^t\|_2^2 \\ &= -(\alpha_t - \frac{L\alpha_t^2}{2}) \|\nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)\|_2^2 + \frac{L\alpha_t^2}{2} \|\xi^t\|_2^2 - (\alpha_t - L\alpha_t^2) \langle \nabla_{\mathbf{w}} \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t), \xi^t \rangle. \end{aligned}$$

Similarly, by the Lipschitz continuity of $\nabla_{\mathbf{v}} F(\mathbf{w}, \mathbf{v})$, we have:

$$\begin{aligned} &\mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^{t+1}) - \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) \le \langle \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t), \mathbf{v}^{t+1} - \mathbf{v}^t \rangle + \frac{L}{2} \|\mathbf{v}^{t+1} - \mathbf{v}^t\|_2^2 \\ &= \langle \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t), -\alpha_t e^t \otimes \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) \rangle + \frac{L}{2} \|\mathbf{v}^{t+1} - \mathbf{v}^t\|_2^2 \\ &= -\alpha_t \langle \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t), e^k \otimes \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) \rangle + \frac{L\alpha_t^2}{2} \|e^t \otimes \nabla_{\mathbf{v}} \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t)\|_2^2. \end{aligned}$$

Note that when G is not used, the bound for $\mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) - \mathbb{F}(\mathbf{w}^t, \mathbf{v}^t)$ is still the same, but the bound for $\mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^{t+1}) - \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t)$ is now simply $\mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^{t+1}) - \mathbb{F}(\mathbf{w}^{t+1}, \mathbf{v}^t) \le 0$.

Combining the above two equations, we then have:

1. *If G is used,*

$$\begin{aligned}
&\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})-\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})=\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})-\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})+\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})-\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\\
&\leq-(\alpha_t-\frac{L\alpha_t^2}{2})\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2+\frac{L\alpha_t^2}{2}\|\xi^t\|_2^2-(\alpha_t-L\alpha_t^2)\langle\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t}),\xi^t\rangle\\
&-\alpha_t\langle\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{k}),e^t\otimes\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\rangle+\frac{L\alpha_t^2}{2}\|e^t\otimes\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2.
\end{aligned}$$

If G is not used,

$$\begin{aligned}
&\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})-\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\leq\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})-\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\\
&\leq-(\alpha_t-\frac{L\alpha_t^2}{2})\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2+\frac{L\alpha_t^2}{2}\|\xi^t\|_2^2-(\alpha_t-L\alpha_t^2)\langle\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t}),\xi^t\rangle
\end{aligned}$$

Taking expectation of both sides and since $\mathbf{E}[\xi^t]=0$*, we have if G is used:*

$$\begin{aligned}
&\mathbf{E}[\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})]-\mathbf{E}[\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})]\\
&\leq-(\alpha_t-\frac{L\alpha_t^2}{2})\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\frac{L\alpha_t^2}{2}\mathbf{E}[\|\xi^t\|_2^2]-\alpha_t\mathbf{E}[\langle\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t}),e^t\otimes\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\rangle]\\
&+\frac{L\alpha_t^2}{2}\mathbf{E}[\|e^t\otimes\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]\\
&=-(\alpha_t-\frac{L\alpha_t^2}{2})\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\frac{L\alpha_t^2}{2}\mathbf{E}[\|\xi^t\|_2^2]-\frac{\alpha_t b}{n}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]\\
&+\frac{L\alpha_t^2}{2}\sum_{i=1}^{n}\mathbf{E}[\|e_i^t\frac{\partial_{v_i}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})}{\partial v_i}\|_2^2]\\
&\leq-(\alpha_t-\frac{L\alpha_t^2}{2})\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\frac{L\alpha_t^2}{2}\mathbf{E}[\|\xi^t\|_2^2]-\frac{\alpha_t b}{n}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]\\
&+\frac{L\alpha_t^2}{2}\sum_{i=1}^{n}\mathbf{E}[\|\frac{\partial_{v_i}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})}{\partial v_i}\|_2^2]\\
&=-\alpha_t\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]-\frac{\alpha_t b}{n}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]\\
&+\frac{L\alpha_t^2}{2}\{\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]+\mathbf{E}[\|\xi^t\|_2^2].\}
\end{aligned}$$

where the second equality follows from $\mathbf{E}[\langle\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t}),e^t\otimes\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\rangle]=\frac{b}{n}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]$*, which can be checked by a straightforward combinatorial argument.*

Following a similar chain of steps, we have the following bound if G is not used: $\mathbf{E}[\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})]-\mathbf{E}[\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})]\leq-\alpha_t\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\frac{L\alpha_t^2}{2}\{\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\mathbf{E}[\|\xi^t\|_2^2].\}$ *Since there are only a finite number of training samples, all the random quantities have bounded support and all the second moments are upper bounded, leading to* $\mathbf{E}[\|\xi^t\|_2^2]<\infty,\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]<\infty,\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]<\infty$*, and let* B *be an upper bound on* $\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]+\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\|_2^2]+\mathbf{E}[\|\xi^t\|_2^2]$*.*

By telescoping, if G is used, we have: $\mathbf{E}[\mathbb{F}(\mathbf{w}^{T+1},\mathbf{v}^{T+1})]-\mathbb{F}(\mathbf{w}^{0},\mathbf{v}^{0})=\sum_{k=0}^{T}\left\{\mathbf{E}[\mathbb{F}(\mathbf{w}^{k+1},\mathbf{v}^{k+1})]-\mathbf{E}[\mathbb{F}(\mathbf{w}^{k},\mathbf{v}^{k})]\right\}\leq-\sum_{k=0}^{T}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{k},\mathbf{v}^{k})\|_2^2]-\sum_{k=0}^{T}\frac{\alpha_k b}{n}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{k+1},\mathbf{v}^{k})\|_2^2]+\frac{LB}{2}\sum_{k=0}^{T}\alpha_k^2$*. Taking the limit* $T\to\infty$ *of both sides, we obtain:*

$$\begin{aligned}
&-\sum_{k=0}^{\infty}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{k},\mathbf{v}^{k})\|_2^2]-\frac{b}{n}\sum_{k=0}^{\infty}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{k+1},\mathbf{v}^{k})\|_2^2]+\frac{LB}{2}\sum_{k=0}^{\infty}\alpha_k^2\\
&\geq\lim_{T\to\infty}\mathbf{E}[\mathbb{F}(\mathbf{w}^{T+1},\mathbf{v}^{T+1})]-\mathbf{E}[\mathbb{F}(\mathbf{w}^{0},\mathbf{v}^{0})]\geq\min_{\mathbf{w},\mathbf{v}}\mathbb{F}(\mathbf{w},\mathbf{v})]-\mathbb{F}(\mathbf{w}^{0},\mathbf{v}^{0})>-\infty.
\end{aligned}$$

Since $\sum_{k=0}^{\infty}\alpha_k^2 < \infty$, *the above inequality immediately implies that* $\sum_{k=0}^{\infty}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^k,\mathbf{v}^k)\|_2^2] < \infty$ *and* $\sum_{k=0}^{\infty}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^{k+1},\mathbf{v}^k)\|_2^2] < \infty$.

If G *is not used, then by a similar argument, we have* $\sum_{k=0}^{\infty}\alpha_k\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^k,\mathbf{v}^k)\|_2^2] < \infty$.

By Lemma A.5 in (Mairal, 2013), to show $\lim_{k\to\infty}\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^k,\mathbf{v}^k)\|_2^2] = 0$, *since* α_k *is not summable, it suffices to show* $\left|\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{k+1},\mathbf{v}^{k+1})\|_2^2] - \mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^k,\mathbf{v}^k)\|_2^2]\right| \le C\alpha_k$ *for some constant* C. *To do so, we first recall a useful fact: for any two vectors* $\mathbf{a},\mathbf{b}$ *and any finite-dimensional vector norm* $\|\cdot\|$,

$$\Big|(\|a\|+\|b\|)(\|a\|-\|b\|)\Big| \le \|a+b\|\|a-b\|. \tag{17}$$

We have:

$$\begin{aligned}
&\left|\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2^2] - \mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2^2]\right| \\
&= \left|\mathbf{E}\Big[\big(\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2 + \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\big)\big(\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2 - \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\big)\Big]\right| \\
&\le \mathbf{E}\left[\left|\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2 + \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\right| \cdot \left|\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2 - \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\right|\right] \\
&\le \mathbf{E}\left[\left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1}) + \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\right\|_2 \cdot \left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t+1})\|_2 - \|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\right\|\right] \\
&\le 2LB\mathbf{E}\Big[\|(\mathbf{w}^{t+1},\mathbf{v}^{t+1})_-(\mathbf{w}^{t},\mathbf{v}^{t})\|_2\Big] \\
&\le 2LB\alpha_t\mathbf{E}\left[\left\|\Big(\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t}) + \xi^k, e^t\otimes\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\Big)\right\|_2\right] \\
&= 2LB\alpha_t\mathbf{E}\left[\sqrt{\left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t}) + \xi^k\right\|_2^2} + \sqrt{\left\|e^t\otimes\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\right\|_2^2}\right] \\
&\le 2LB\alpha_t\sqrt{\mathbf{E}\left[\left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t}) + \xi^k\right\|_2^2\right] + \mathbf{E}\left[\left\|e^t\otimes\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\right\|_2^2\right]} \\
&\le 2LB\alpha_k\sqrt{2\mathbf{E}\left[\left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t},\mathbf{v}^{t})\right\|_2^2\right] + 2\mathbf{E}\left[\left\|\xi^t\right\|_2^2\right] + \mathbf{E}\left[\left\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^{t+1},\mathbf{v}^{t})\right\|_2^2\right]} \\
&\le 2LB\alpha_t\sqrt{5B^2} = 2\sqrt{5}B^2L\alpha_t,
\end{aligned}$$

where the first inequality is an application of Jensen's inequality, the second inequality follows from Equation (17) *the third inequality follows from Lipschitz continuity and the sixth inequality follows from another application of Jensen's inequality. Consequently, by Lemma and the above chain of inequalities, it follows that* $\lim_{t\to\infty}\mathbf{E}[\|\nabla_{\mathbf{w}}\mathbb{F}(\mathbf{w}^t,\mathbf{v}^t)\|_2^2] = 0$. *Note that if* G *is used, then by a similar argument, it follows that* $\lim_{t\to\infty}\mathbf{E}[\|\nabla_{\mathbf{v}}\mathbb{F}(\mathbf{w}^t,\mathbf{v}^t)\|_2^2] = 0$. *Consequently, if* G *is used, then* $\lim_{t\to\infty}\mathbf{E}[\|\nabla\mathbb{F}(\mathbf{w}^t,\mathbf{v}^t)\|_2^2] = 0$.

C. Proof of Proposition 1

Proposition 1 *Suppose* $(\mathbf{x},y)$ *denotes a training sample and its corrupted label. For simplicity, let the MentorNet input* $\phi(\mathbf{x},y,\mathbf{w}) = \ell$ *be the loss computed by the StudentNet model parameter* $\mathbf{w}$. *The MentorNet* $g_m(\ell;\Theta) = v$, *where* v *is the sample weight. If* g_m *decreases with respect to* ℓ, *then there exists an underlying robust objective* F:

$$F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\rho(\ell_i),$$

where $\rho(\ell_i) = \int_0^{\ell_i} g_m(x;\Theta)dx$. *In the special cases,* $\rho(\ell)$ *degenerates to the classical robust M-estimator: Huber (Huber et al., 1964) and the log-sum penalty (Candes et al., 2008).*

Proof 2 *Given a MentorNet g_m and its fixed parameter Θ. As $\phi(\mathbf{x}, y, \mathbf{w}) = \ell$, we have $g_m(\phi(\mathbf{x}, y, \mathbf{w}); \Theta) = g_m(\ell; \Theta)$.*

g_m is a neural network and hence is continuous with respect to ℓ. Define $\rho(\ell)$ as

$$\rho(\ell) = \int_0^{\ell} g_m(x; \Theta) dx \tag{18}$$

Given the condition in the proposition, g_m is decreasing with respect to ℓ. The function ρ is then bounded by its 1st term of the Taylor series about a point $\mathbf{w}^*$. We have:

$$\rho(\phi(\mathbf{x}, y, \mathbf{w})) \leq \rho(\phi(\mathbf{x}, y, \mathbf{w}^*)) + g_m(\phi(\mathbf{x}, y, \mathbf{w}^*); \Theta)(\phi(\mathbf{x}, y, \mathbf{w}) - \phi(\mathbf{x}, y, \mathbf{w}^*)) \tag{19}$$

According to (Meng et al., 2015), the right-hand side in Eq. (19) is a tractable surrogate for $\rho(\phi(\mathbf{x}, y, \mathbf{w}))$ and there exists an underlying robust objective. For the i-th sample, we have:

$$\rho(\phi(\mathbf{x}_i, y_i, \mathbf{w})) = \rho(\ell_i) = \int_0^{\ell_i} g_m(x; \Theta) dx \tag{20}$$

Finally, we have a robust objective derived from:

$$F(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \rho(\phi(\mathbf{x}_i, y_i, \mathbf{w})) = \frac{1}{n} \sum_{i=1}^{n} \rho(\ell_i) \tag{21}$$

Now we show the connection between Eq. (20) to the robust M-estimator. For simplicity, we assume that the loss $\ell \geq 0$ is non-negative for every training sample. For the Huber loss, there exists an Θ^* such that:

$$g_m(\ell; \Theta^*) = \begin{cases} \frac{1}{2} & \ell \leq \lambda^2 \\ \frac{\lambda}{2\sqrt{\ell}} & \textit{otherwise} \end{cases}, \tag{22}$$

where $\lambda > 0$. It is easy to verify g_m is decreasing with respect to ℓ, and we have:

$$\rho(\ell) = \int_0^{\ell} g_m(x; \Theta^*) dx = \begin{cases} \frac{1}{2}\ell & \ell \leq \lambda^2 \\ \lambda(\sqrt{\ell} - \frac{1}{2}\lambda) & \textit{otherwise} \end{cases} \tag{23}$$

Therefore we have:

$$\rho(\ell^2) = \begin{cases} \frac{1}{2}\ell^2 & \ell \leq \lambda^2 \\ \lambda(\ell - \frac{1}{2}\lambda) & \textit{otherwise} \end{cases} \tag{24}$$

Eq. (24) has a similar form of the Huber M-estimator (Huber et al., 1964).

Likewise, there exists an Θ^* and a positive ϵ such that

$$g_m(\ell; \Theta^*) = \frac{\lambda}{\ell + \epsilon} \tag{25}$$

Its underlying objective is identical to the log-sum penalty (Candes et al., 2008):

$$\rho(\ell) = \int_0^{\ell} g_m(x; \Theta^*) dx = \lambda \log(\ell + \epsilon) - \lambda \log(\epsilon) = \lambda \log(1 + \frac{\ell}{\epsilon}) \tag{26}$$

It leads to the following log-sum penalty:

$$F(\mathbf{w}) = \lambda \frac{1}{n} \sum_{i=1}^{n} \log(1 + \frac{\ell_i}{\epsilon}) \tag{27}$$

For the Lorentzian (Black & Anandan, 1996). We have

$$g_m(\ell;\Theta^*) = \frac{2\ell}{2\delta^2+\ell^2} \tag{28}$$

In special cases, we assume that δ is a small positive such that all sample loss $\ell \geq \sqrt{2}\delta$, the underlying objective is

$$\rho(\ell) = \int_0^{\ell} g_m(x;\Theta^*)dx = \log(2\delta^2+\ell^2) - \log 2\delta^2 = \log(1+\frac{1}{2}(\frac{\ell}{\delta})^2) \tag{29}$$

The above objective becomes Lorentzian (Black & Anandan, 1996), also known as Lorentzian/Cauchy.

D. Comparison on MentorNet Architectures

This section examines *MentorNet*'s architecture in terms of learning existing predefined curriculums (or weighting scheme). To generate the data for training MentorNet, we enumerate the feasible input space of $\mathbf{z}_i$ including the loss, difference to the loss moving average, label, and epoch percentage. For this experiment, the dataset contains a total of 300k samples. For each sample, we label a weight according to the weighting scheme in the predefined curriculum. For example, we compute the weight for focal loss by

$$v_i^* = [1-\exp\{-\ell_i\}]^{\gamma}, \tag{30}$$

where γ is a hyperparameter for smoothing the distribution. We consider the following predefined curriculums: self-paced (Kumar et al., 2010), hard negative mining (Felzenszwalb et al., 2010), linear weighting (Jiang et al., 2015), focal loss (Lin et al., 2017), and prediction variance (Chang et al., 2017). Besides, we also consider a temporal mixture weighting which is a mix of the self-paced (when the epoch percentage < 50) and the hard negative mining (when the epoch percentage ≥ 50). In some curriculums, the form G is unknown. We directly minimize the mean squared error between the MentorNet's output and the true weight.

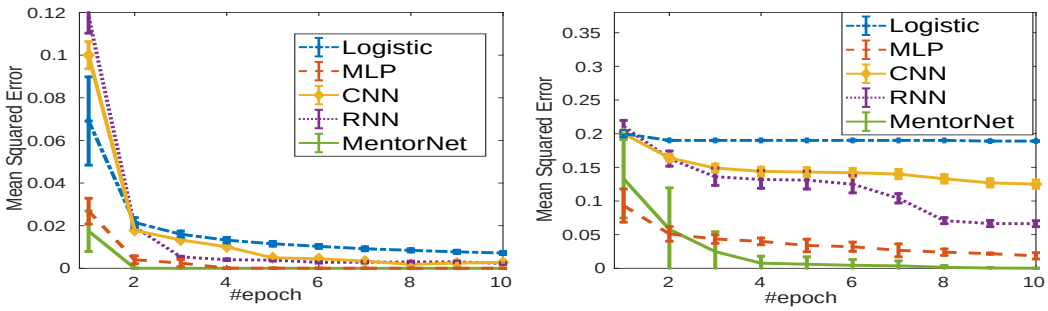
We compare the MentorNet architecture in Fig. 1 in the paper to a simple logistic regression and 3 classical networks: a 2-layer Multiple Layer Perceptron (MLP), a 2-layer CNN with mean pooling, and an LSTM network (RNN) to sequentially encode the features of every example in a mini-batch. The same features are used across all networks. The objective is minimized by the Adam optimizer (Kingma & Ba, 2015). We use a very simple network for the proposed MentorNet. The bidirectional LSTM has a single layer of 10 base LSTM units (step size = 10). We use an embedding layer (size = 2) for the labels and an embedding layer (size = 5) for the integer epoch percentage between 0 and 99. The fully connected layer fc_1 uses the tangent activation function and has 20 hidden nodes.

The performance is evaluated by the Mean Squared Error (MSE) to the true weight produced by each curriculum. Each experiment is repeated 5 times using random starting values, and the average MSE (with the 90% confidence interval) is reported. Table 1 shows the comparison results. As we see, the simple network structure MLP performs well except for complex weighting schemes that are prediction variance (Chang et al., 2017) and Temporal mixture weighting. Nevertheless, the bi-LSTM structure in Fig.1 of the paper performs better than other classical network architectures. Fig. 1 illustrates the error curve of different MentorNet architecture during training, where the x-axis is the training epoch and y-axis is the MSE.

Table 1. The MSE comparison of different MentorNet architecture on predefined curriculums.

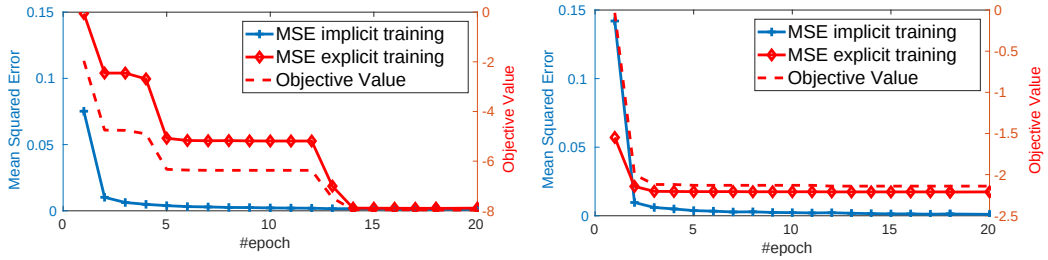
Weighting Scheme	Logistic	MLP	CNN	RNN	MentorNet (bi-LSTM)
Self-paced (Kumar et al., 2010)	8.9±0.8E-3	1.1±0.3E-5	4.9±1.0E-2	1.6±1.0E-2	**1.6±0.5E-6**
Hard negative mining (Felzenszwalb et al., 2010)	7.1±0.7E-3	1.6±0.6E-5	2.7±0.6E-3	2.2±0.4E-3	**6.6±4.5E-7**
Linear weighting (Jiang et al., 2015)	9.2±0.1E-4	1.2±0.4E-4	1.1±0.2E-4	2.0±0.3E-2	**4.4±1.3E-5**
Prediction variance (Chang et al., 2017)	6.8±0.1E-3	4.0±0.1E-3	2.8±0.4E-2	6.2±0.2E-3	**1.4±0.7E-3**
Focal loss (Lin et al., 2017)	1.7±0.0E-3	6.0±3.5E-5	1.2±0.3E-2	1.2±0.3E-2	**1.5±0.8E-5**
Temporal mixture weighting	1.8±0.0E-1	1.9±0.4E-2	1.2±0.6E-1	6.6±0.4E-2	**1.2±1.1E-4**

In Table 1, we learn a MentorNet by minimizing the MSE between its output and the true weight. We call it implicit training. When the form of G is known, we can learn a MentorNet by directly minimizing Eq.(4) in the paper, called explicit training. These two training approaches are theoretically identical and we empirically compare them on two curriculums of known G in Fig. 2. As shown, we found implicit training converges faster.



(a) Hard negative mining (b) Temporal mixture weighting

Figure 1. Comparison of explicit and implicit MentorNet training.



(a) Self-paced (b) Linear weighting

Figure 2. Convergence comparison of different MentorNet architectures.

E. Implementation Details

E.1. Dataset and StudentNet

CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009) consist of 32 $\times$ 32 color images arranged in 10 and 100 classes. Both datasets contain 50,000 training and 10,000 validation images. The inception (Szegedy et al., 2016) and wide-resnet-101 (He et al., 2016; Zagoruyko & Komodakis, 2016) are used as the StudentNet. Their implementations are based on the TensorFlow slim implementation[1], where the inception is detailed in (Zhang et al., 2017) and the wider resnet is in (Zagoruyko & Komodakis, 2016).

In training, the batch size is set to 128 and we train 39K iterations for resnet model and 120k for the inception model. The Step 12 in Algorithm 1 in the paper is implemented by the momentum SGD optimizer (momentum = 0.9) on a single GPU. We use the common learning rate scheduling strategy and set the starting learning rate as 0.1 and multiply it by a factor of 0.1 at the 19.5k, 25k and 30k iteration for the resnet model, and 78k for the inception model. Training in this way on the clean training dataset, the validation accuracy reaches about 81.4% and 95.5% for inception and resnet-101 on CIFAR-10, and about 49.2% and 78.8% on CIFAR-100.

By default, the StudentNet incorporates three types of regularization: 1) the weight decay which adds an l_2 norm of the model parameters into the learning objective; 2) data augmentation (Krizhevsky et al., 2012) which augments the training images via domain-specific transformations random cropping, perturbation and contrast; 3) dropout (Srivastava et al., 2014) masks out network fully-connected layer outputs randomly. We use the best hyperparameter found on the clean training data. In the inception network, weight decay is set $4e{-}3$ and the dropout keep probability is 0.5. In the resnet-101, weight decay is $2e{-}4$ and the dropout keep probability is 1.0. In the Forgetting baseline (Arpit et al., 2017), we search the dropout parameter in the range of (0.2-0.9) and report the best accuracy on the validation set. The data augmentation is the same for the inception and the resnet-101: we pad 4 pixels to each side and randomly sampling a 32 x 32 crop, randomly flip the image horizontally (left to right), and then linearly scale the image to have zero mean and unit norm. Unless specified otherwise, the StudentNet of the same hyperparameter, discussed above, is used in all baseline and the proposed model.

ImageNet ILSVRC2012 (Deng et al., 2009) contains about 1.2 million training and 50k validation images, split into 1,000 classes. Each image is resized to 299 $\times$ 299 with 3 color channels. For the ImageNet, we use the inception-resnet v2 slim

[1]https://github.com/tensorflow/models/tree/master/research/slim

implementation[2] as our StudentNet. We train the model on the ImageNet of 40% noise. The Step 12 in Algorithm 1 in the paper is implemented as a distributed asynchronized momentum SGD optimizer (momentum = 0.9) on 50 GPUs. We set the batch size to 32 and train the model until it converges. That is 500 thousand steps for the StudentNet without any regularization and 1 million steps for the StudentNet with full regularization (weight decay, dropout and data augmentation). The starting learning rate is 0.05 and is decreased by a factor of 0.1 every 10 training epochs. The default dropout-keep-probability hyperparameter set to 0.8. In the Forgetting baseline, we set it to 0.2, which is the best parameter found on CIFAR-100. The weight decay is $4e{-}5$. The batch norm is used and its decay is set to 0.9997 and the epsilon is set to 0.001. The default data augmentation in the slim implementation is used. Training in this way on the clean training dataset, the validation accuracy is Hit@1=0.765.

E.2. Baselines

Regarding the baseline method. FullMode is the standard StudentNet trained using l_2 weight decay, dropout (Srivastava et al., 2014) and data augmentation (Krizhevsky et al., 2012). The same parameters discussed in Appendix E.1 are used. Forgetting was introduced in (Arpit et al., 2017). It is same as the FullModel except that the dropout parameter is tuned in the range of (0.2-0.9). For the self-paced learning (Kumar et al., 2010), we gradually increase λ in training by 20%. Following (Jiang et al., 2014) we tune the parameter in the range of the 50th, 60th and 75th percentile of average sample loss. For the focal loss (Lin et al., 2017), we tune its γ in the range of $\{1, 2, 3\}$ in our experiments. It is easy to verify that Eq. (30) leads to the same classification objective in the focal loss (Lin et al., 2017), an award-winning method for object detection. For the Reed (Reed et al., 2014), we implement two versions: the soft and hard version. Let $q = [q_1, \ldots, q_m]$ be the prediction (logits after softmax) for m classes:

$$\ell_i = -(\beta \sum_{j=1}^{m} \mathbb{1}(y_i = j) \log(q_j) + (1 - \beta) \sum_{j=1}^{m} q_j \log(q_j)), \tag{31}$$

where $\mathbb{1}$ is the indicator function and a hard version:

$$\ell_i = -(\beta \sum_{j=1}^{m} \mathbb{1}(y_i = j) \log(q_j) + (1 - \beta) \max_j \log(q_j)), \tag{32}$$

We tune the parameters β in the range of $\{0.7, 0.8, 0.9, 0.95\}$. Goldberger (Goldberger & Ben-Reuven, 2017) is a recent baseline weakly-supervised learning method. We implement the S-Model in the paper by appending an additional layer to the StudentNet.

E.3. Setups of Our Model

The details about the MentorNet architecture (Fig. 1 in the paper) are discussed in Appendix D. MentorNet PD is learned using the curriculum in Eq. (5) in the paper. The loss moving average ℓ_{pt} is set to the 75th-percentile loss in a mini-batch. We tune the hyperparameter λ_1 and λ_2. MentorNet DD is the learned *data-driven* curriculum. It is trained on 5,000 images of true labels, randomly sampled from CIFAR-10. We learn MentorNet DD on this dataset and apply it to CIFAR-100 on which no true labels are used. We use the CIFAR-10 subset of the same level of the noise fraction corresponding to the CIFAR-100. CIFAR-10 and CIFAR-100 are two different datasets that have not only different classes but also the different number of classes. As CIFAR-100 and CIFAR-10 have the different number of classes, to apply a MentorNet, we fix the class label to 0.

Algorithm 1 is used to train the MentorNet together with the StudentNet. The decay factor in computing the loss moving average is set to 0.95. As mentioned in the paper, a burn-in process is used in the first 20% training epoch for both MentorNet DD and MentorNet PD. We update and learn MentorNet twice after the learning rate is changed. That is on the 21% and 75% of the total epoch. More updates lead to insignificant performance difference. For each mini-batch, the weight decay parameter θ in Eq. (1) in the paper is normalized by the sum of the weight in a mini-batch. That is $\theta^t = \frac{1}{b}\theta^0 \sum_{i=1}^{b} \mathbf{v}_{\Xi_i}$, where θ^0 is the original weight decay parameter. The same learning rate scheduling strategy in the StudentNet is used in Algorithm 1.

[2] https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_resnet_v2.py

References

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

Black, M. J. and Anandan, P. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer vision and image understanding*, 63(1):75–104, 1996.

Candes, E. J., Wakin, M. B., and Boyd, S. P. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

Chang, H.-S., Learned-Miller, E., and McCallum, A. Active bias: Training a more accurate neural network by emphasizing high variance samples. *NIPS*, 2017.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

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

Gong, P., Zhang, C., Lu, Z., Huang, J. Z., and Ye, J. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *ICML*, 2013.

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

Huber, P. J. et al. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

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

Jiang, L., Meng, D., Zhao, Q., Shan, S., and Hauptmann, A. G. Self-paced curriculum learning. In *AAAI*, 2015.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

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

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. *ICCV*, 2017.

Mairal, J. Stochastic majorization-minimization algorithms for large-scale optimization. In *NIPS*, 2013.

Meng, D., Zhao, Q., and Jiang, L. What objective does self-paced learning indeed optimize? *arXiv preprint arXiv:1511.06049*, 2015.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

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, C.-H. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, pp. 894–942, 2010.